

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
вычислительной техники

Кафедра
МОиПЭВМ

Направление подготовки 09.03.04 «Программная инженерия»

Направленность Программная инженерия

БАКАЛАВРСКАЯ РАБОТА

на тему

**Мобильное приложение для конструирования и
выполнения практических занятий школьного
курса химии**

Студент

(подпись, дата)

Садыков Захар Ринатович

(ФИО полностью)

Руководитель

(подпись, дата)

Балашова И.Ю.

(фамилия, инициалы)

Нормоконтролёр

(подпись, дата)

Такташкин Д.В.

(фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой

(подпись)

Козлов А.Ю.

(фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК

(подпись)

Попова Н.А.

(фамилия, инициалы)

Пенза, 2024

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
Вычислительной техники

Кафедра
МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
_____ А.Ю. Козлов
«25» декабря 2023 г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент Садыков Захар Ринатович гр. 20ВП1 факультета ВТ
<Фамилия, имя, отчество полностью>

2. Тема работы Мобильное приложение для конструирования и выполнения
практических занятий школьного курса химии

Тема утверждена приказом ПГУ № _____ от " _____ " _____ 20 _____

3. Руководитель работы _____ к.т.н., доцент кафедры «МО и ПЭВМ» Балашова И.Ю.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

1) *Назначение: мобильное приложение для конструирования и выполнения*
практических занятий школьного курса химии.

2) *Среда выполнения: операционная система Windows 10*

Языковые средства разработки: C#, UML

Инструментальные средства: IntelJ Rider, Unity, Draw.io,

Технологии разработки: ООП, принципы SOLID, шаблоны проектирования.

3) *Основные функции приложения:*

а) авторизация пользователя;

б) регистрация пользователя;

в) проведение химического опыта учеником;

г) просмотр прогресса пользователя;

.

5. Перечень подлежащих разработке вопросов:

1) *Анализ предметной области и требований к системе;*

2) <i>Проектирование программных средств;</i>
3) <i>Реализация программных средств;</i>
4) <i>Тестирование программных средств;</i>
5) <i>Оформление пояснительной записки.</i>

5. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и требований к системе	10%	08.04.2024 – 13.04.2024	
2	Проектирование программных средств	20%	15.04.2024 – 25.04.2024	
3	Реализация программных средств	30%	26.04.2024 – 12.05.2024	
4	Тестирование программных средств	20%	13.05.2024 – 22.05.2024	
5	Оформление пояснительной записки	20%	23.05.2024 – 03.06.2024	

Дата выдачи задания «21» декабря 2023г.

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить
Декан ФВТ _____ Л.Р. Фионова
(подпись, дата)

Реферат

Пояснительная записка содержит 79 листов, 41 рисунок, 14 таблиц и 20 использованных источников.

ПРАКТИЧЕСКИЕ РАБОТЫ ПО ХИМИИ, ДОПОЛНЕННАЯ РЕАЛЬНОСТЬ, БАЗА ДАННЫХ, МОБИЛЬНОЕ ПРИЛОЖЕНИЕ.

Объект исследования – мобильные приложения для конструирования практических занятий школьного курса химии в дополненной реальности.

Цель выпускной квалификационной работы – спроектировать и разработать мобильное приложения, которое позволяет конструировать и выполнять практические занятия по школьному курсу химии.

Технологии разработки – разработка визуализации химических реакций в среде разработки IntelJ Rider 2023 на объектно-ориентированном языке программирования C#. В качестве основной платформы используется Unity, как средство визуализации. Развертывание основного приложения осуществляется на операционных системах семейства Android и IOS. В рамках обработки и хранения данных приложение взаимодействует с сервисами Firebase Authentication, Realtime Database, Storage. В качестве среды исполнения программных средств выступает мобильное устройство пользователя.

Результаты работы – в ходе выполнения выпускной квалификационной работы спроектировано и разработано мобильное приложения, которое позволяет конструировать и выполнять практические занятия по школьному курсу химии.

					ПГУ 09.03.04– 8ВР201.24 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата	Мобильное приложение для конструирования и выполнения практических занятий школьного курса химии. Пояснительная записка	Лит.	Лист	Листов
Разраб.		Садыков З.Р.						
Пров.		Балашова И.Ю.					4	79
						Группа 20ВП1		
Н. Контр.		Такташкин Д.В.						
Утв.		Козлов А.Ю.						

Содержание

Введение.....	7
1 Анализ инструментов для выполнения школьных практических работ по химии.....	8
1.1 Анализ предметной области и постановка задачи	8
1.2 Анализ программного обеспечения, направленного на проведение практических химических работ	9
1.2.1 Мобильное приложение «AR Chemistry Lab».....	9
1.2.2 Мобильное приложение «Unreal Chemist».....	10
1.2.3 Мобильное приложение «Elements 4D»	11
1.2.4 Мобильное приложение «AR Chemist- Science Lab in AR»	12
1.2.5 Сравнение мобильных приложений выполнения практических работ по химии	13
1.3 Анализ функциональных требований.....	14
1.4 Анализ нефункциональных требований.....	16
1.4.1 Unity	16
1.4.2 Firebase Authentication	17
1.4.3 Firebase Realtime Database.....	17
1.4.4 Firebase Storage.....	17
1.4.5 ARCore	18
1.5 Планирование разработки и оценка бюджета	18
2 Проектирование мобильного приложения выполнения школьных лабораторных работ по химии.....	21
2.1 Архитектура приложения.....	21
2.2 Проектирование структуры приложения	22
2.2.1 Концептуальная модель базы данных	22
2.2.2 Логическая модель базы данных.....	23
2.3 Проектирование пользовательского интерфейса	23
3 Реализация приложения для проведения школьных практических работ по химии	34

3.1 Реализация базы данных	34
3.2 Реализация пользовательского интерфейса	36
3.3 Реализация состояния переходов в приложении	36
3.4 Реализация программных средств.....	37
4 Контроль качества программного обеспечения.....	38
4.1 Тестирование черным ящиком.....	38
4.1.1 Классы эквивалентности	38
4.1.2 Граничные условия.....	39
4.2 Модульные тесты	40
4.3 Метрики кода	43
4.3.1 Метрика SLOC	43
4.3.2 Метрика Холстеда.....	43
Заключение	46
Список использованных источников	47
Приложение А (рекомендуемое) Листинг приложения.....	49
Приложение А (рекомендуемое) Результаты реализации интерфейса	70
Приложение В (рекомендуемое) Таблица диаграммы класса.....	76

Введение

В современном образовательном процессе использование информационных технологий становится все более значимым элементом. Одной из областей, где цифровые инструменты могут существенно улучшить эффективность обучения, является проведение школьных практических работ по химии.

Химия – это предмет, требующий от учащихся не только теоретических знаний, но и практических навыков. Однако проведение экспериментов в классе может быть ограничено доступностью специализированного оборудования и химических реагентов, а также временными ограничениями. В результате, ученики могут лишь наблюдать и запоминать процессы, не получая возможности для собственных исследований и экспериментов.

В свете этих проблем разработка мобильного приложения для проведения школьных практических работ по химии представляется актуальной и перспективной задачей. Такое приложение может предоставить учащимся возможность самостоятельно проводить виртуальные эксперименты, изучать химические процессы и получать практические навыки, не выходя из класса или дома.

Целью выпускной квалификационной работы является разработка мобильного приложения, предоставляющего возможность учащимся проводить виртуальные практические работы по химии. Разработка включает в себя анализ требований, проектирование, реализацию и тестирование.

1. Анализ инструментов для выполнения школьных практических работ по химии

1.1 Анализ предметной области и постановка задачи

Химия, как наука о веществах, их свойствах и превращениях, играет ключевую роль в формировании базовых знаний школьников и развитии их понимания мира вокруг. В свете современных технологий и цифровизации образования возникает необходимость в инновационных подходах к обучению химии, которые бы сочетали в себе следующие аспекты:

- интерактивность;
- вовлеченность;
- удобство;
- доступность.

Часто абстрактные концепции химии могут вызывать трудности в усвоении и понимании учащихся. В связи с чем подход к вопросу об изучении материала можно сделать более практическим путем того, что ученики самостоятельно делают все шаги по выполнению работы. Использование приложения не только даст ученикам возможность экспериментировать и применять теоретические знания на практике, но и сделать процесс изучения химии увлекательным, интерактивным и эффективным.

Кроме того, не в каждой школе есть достаточное количество оборудования для того, чтобы проводить химические опыты в рамках лабораторных работ. Мобильное же устройство имеется почти у каждого человека, в связи с чем возникает больше возможностей для проведения данных работ. Это является аспектом доступности.

Благодаря приложению у ученика есть возможность выполнить практическую работу во внеурочное время, тем самым дополнительно закрепив материал. Ученик может сделать ее, даже если будет отсутствовать на уроке по причине болезни, что является достаточно удобным как для ученика, так и для учителя, ведь ученик отрабатывает материал.

В этом контексте возникает идея разработки мобильного приложения, предназначенного для конструирования и выполнения практических занятий по химии.

Цель – создать полезный инструмент, который поможет школьникам не только понять химические концепции, но и полюбить процесс изучения этой увлекательной науки.

1.2 Анализ программного обеспечения, направленного на проведение практических химических работ

1.2.1 Мобильное приложение «AR Chemistry Lab»

Приложение «AR Chemistry Lab» является одним из немногих примеров мобильного приложения по химии с дополненной реальностью. Интерфейс приложения продемонстрирован на рисунке 1.



Рисунок 1 – Интерфейс приложения «AR Chemistry Lab»

Среди достоинств данного приложения можно выделить то, что оно предоставляет пользователю возможность проводить разные химические опыты в дополненной реальности, регулировать количество добавляемых реагентов, визуализировать химические реакции различных химических реакций для лучшего понимания процессов, а также дает возможность совместного использования приложения для обучения в классе или дома.

Главным недостатком, с точки зрения пользователя, можно считать платное распространение приложения и поддержка только одной операционной системы.

1.2.2 Мобильное приложение «Unreal Chemist»

Мобильное приложение «Unreal Chemist» предоставляет пользователю возможность проводить химические опыты с множеством возможностей воздействия на реагенты без дополненной реальности. Интерфейс приложения продемонстрирован на рисунке 2.



Рисунок 2 – Интерфейс приложения «Unreal Chemist»

Среди достоинств этого приложения можно сказать, что оно доступно на двух основных мобильных операционных системах (IOS и Android), в связи с чем является достаточно популярным.

Основными недостатками является то, что в приложении использовано множество рекламы, в связи с чем пропадает заинтересованность и вовлеченность, множество ошибок, на которые жалуются пользователи, а также отсутствие дополненной реальности.

1.2.3 Мобильное приложение «Elements 4D»

Приложение «Elements 4D» среди других программных обеспечений подобной тематики, но при этом актуально и по настоящее время. Интерфейс приложения продемонстрирован на рисунке 3.

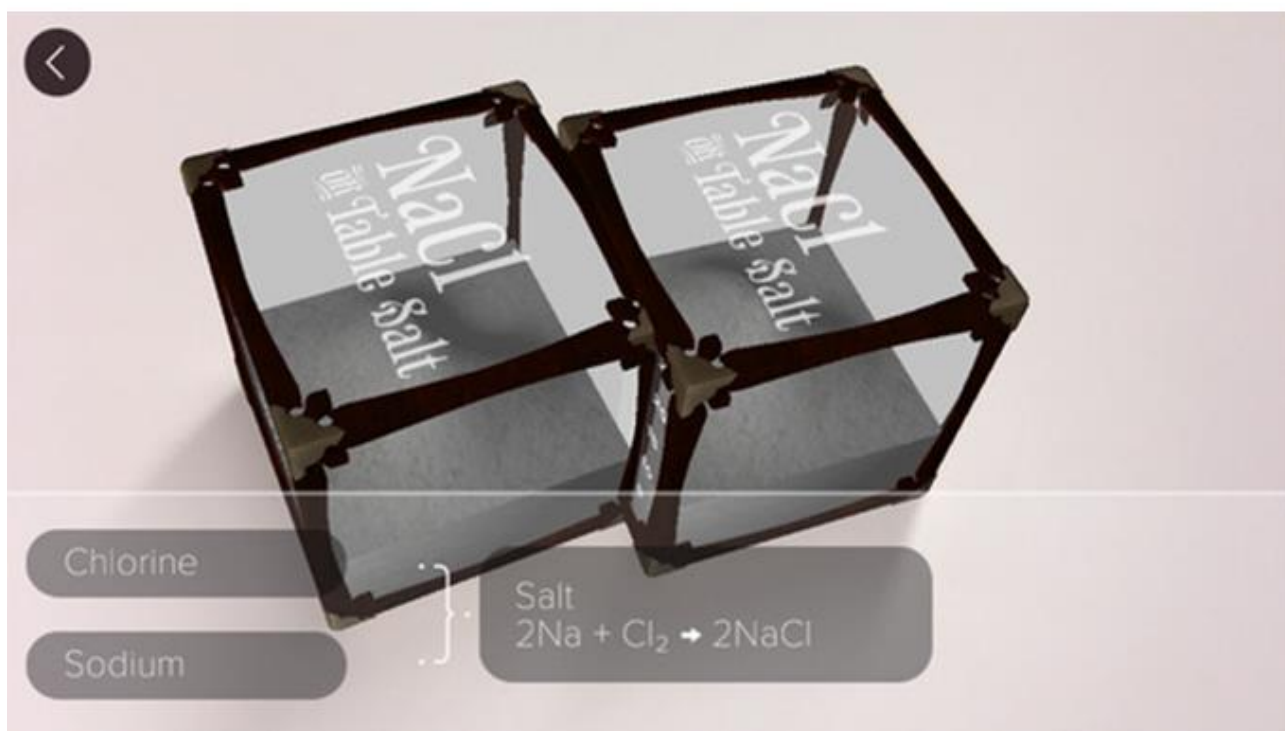


Рисунок 3 – Интерфейс приложения «Elements 4D»

Главным достоинством приложения является то, что оно поддерживается до сих пор и при этом доступно на каждой платформе, а также имеет интересную форму взаимодействия с пользователем.

Из недостатков можно отметить, для работы с приложением необходимы дополнительные специальные карточки, которые помогают приложению ориентироваться в пространстве, что может быть достаточно неудобным.

1.2.4 Мобильное приложение «AR Chemist- Science Lab in AR»

Приложение «AR Chemist- Science Lab in AR» – мобильное программное обеспечение для проведения химической работы с дополненной реальностью. Интерфейс приложения продемонстрирован на рисунке 4.



Рисунок 4 – Интерфейс приложения «AR Chemist- Science Lab in AR»

Приложение является одним из немногих, которое работает на операционной системе Android и при этом является программным обеспечением, работающим с дополненной реальностью, что является весомым достоинством.

К недостаткам можно отнести отсутствие поддержки приложения и несовместимости с новыми версиями Android.

1.2.5 Сравнение мобильных приложений выполнения практических работ по химии

Проведем сравнительный анализ рассмотренных выше программных средств. Для оценки приложений будет использоваться система оценивания, в которой наличие критерия – 1 балл, отсутствие критерия – 0 баллов. Результаты сравнения мобильных приложений практических химических работ в таблице 1.

Таблица 1 – Сравнение программных средств аналогичного назначения

Критерий \ Приложение	AR Chemistry Lab	Unreal Chemist	Elements 4D	AR Chemist- Science Lab in AR
1. Поддержка новых версий Android	0	1	1	0
2. Поддержка новых версий IOS	1	1	1	0
3. Дополненная реальность	1	0	1	1
4. Школьная программа	1	0	0	0
5. Поддержка русского языка	0	0	0	0
6. Бесплатное распространение	0	1	1	1
7. Проверка знаний и их отслеживание	1	0	0	0
ИТОГО	4	2	4	2

Подводя итог, можно сказать, что среди рассмотренных приложений наиболее лучшими являются «AR Chemistry Lab» и «Elements 4D». Из плюсов, которые есть в приложениях можно отметить: дополненную реальность, присутствие школьной программы, а также проверка знаний. При этом у них есть минусы в виде несовместимости со всеми операционными системами для мобильных устройств, отсутствие поддержки русского языка, постоянная реклама, которая отвлекает пользователя.

В результате принято решение разработать собственное программное средство, которое будет включать в себя лучшие решения от рассмотренных приложений и будет лишено наиболее существенных недостатков.

Исходя из данной таблицы, к разрабатываемому приложению были выдвинуты следующие дополнительные требования:

- поддержка русского языка;

- отсутствие платного распространения;
- поддержка популярных мобильных операционных систем.

1.3 Анализ функциональных требований

Важной частью разработки программного обеспечения является необходимость выделить требования к программному продукту [1]. Диаграмма вариантов использования задействована в качестве визуализации функциональных требований к программному продукту. Благодаря ей, можно понять, какие возможности есть у пользователя в программном продукте. В ходе разработки приложения были выявлены следующие функции необходимые для реализации, которые представлены на рисунке 5.

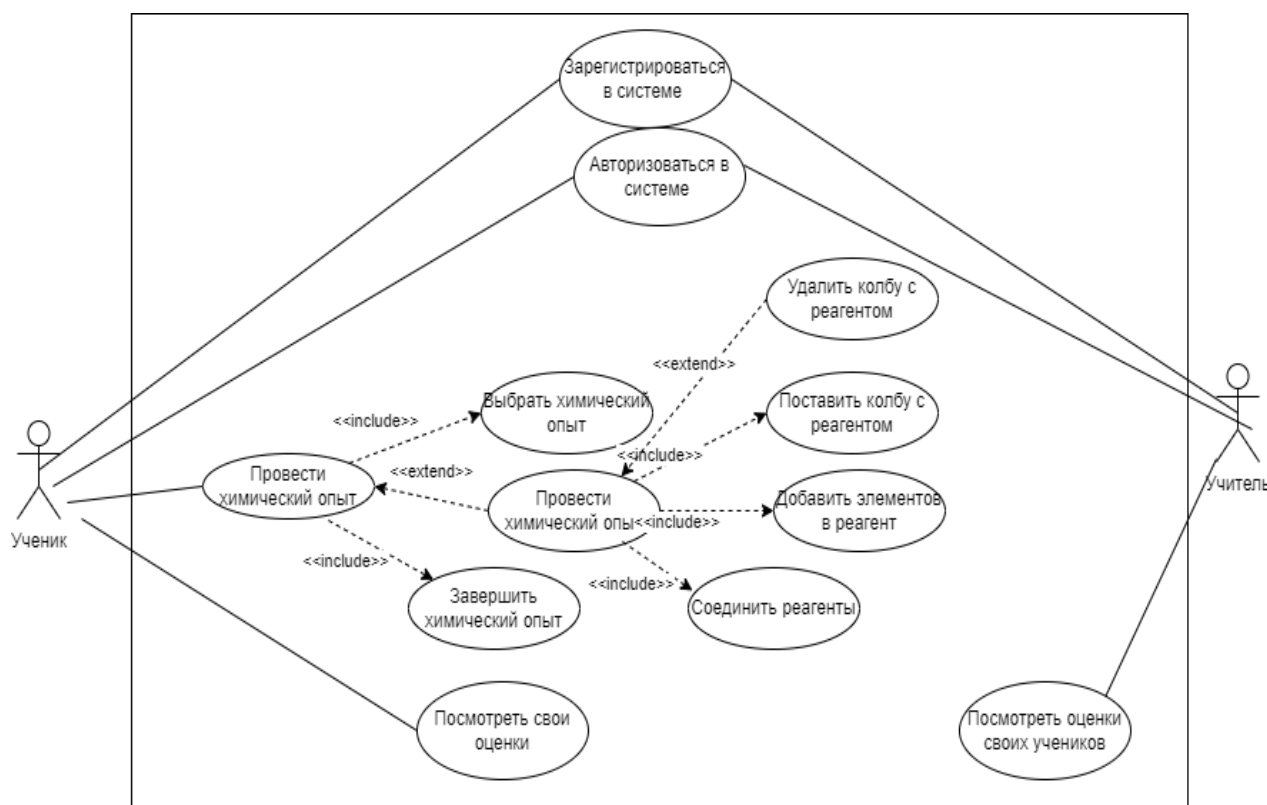


Рисунок 5 – Диаграмма вариантов использования

Основными лицами, участвующими в использовании приложения, являются ученик и учитель. Каждый пользователь имеет возможность авторизоваться и зарегистрироваться в системе. Помимо этого, ученик может провести практическую работу по химии и посмотреть оценки за пройденные уроки в приложении. Учитель же может отслеживать прогресс каждого ученика,

который за ним закреплен. Спецификация для варианта использования «Провести химический опыт» приведена на рисунке 6.

Прецедент: провести химический опыт
ID: 1
Краткое описание: Пользователь проводит полноценный химический опыт
Главные актеры: Система, ученик
Второстепенные актеры: Нет
Предусловие: У ученика есть токе авторизации
Основной поток: 1. Если пользователь авторизован, он выбирает урок, который хочет пройти 2. Система находит нужный урок и загружает данные о необходимых химикатах для эксперимента 3. Пользователь может поставить колбы с химикатами и смешивать для получение результата 4. Пользователь завершает урок и отправляет данные с результатами в БД
Постусловия: Нет
Альтернативные потоки: Нет

Рисунок 6 – Спецификация варианта использования «Провести химический опыт»

Спецификация для варианта использования «Зарегистрироваться в системе» приведена на рисунке 7.

Прецедент: Зарегистрироваться в системе
ID: 2
Краткое описание: Пользователь проходит регистрацию
Главные актеры: Ученик, учитель
Второстепенные актеры: Нет
Предусловие: Нет
Основной поток: 1. Пользователь вводит данные и отправляет их
Постусловия: Нет
Альтернативные потоки: Нет

Рисунок 7 – Спецификация варианта использования «Зарегистрироваться в системе»

Спецификация для варианта использования «Авторизоваться в системе» приведена на рисунке 8.

Прецедент: Авторизация в системе
ID: 3
Краткое описание: Пользователь проходит авторизацию
Главные актеры: Ученик, учитель
Второстепенные актеры: Нет
Предусловие: Нет
Основной поток: 1. Пользователь вводит логин и пароль
Постусловия: Нет
Альтернативные потоки: Нет

Рисунок 8 – Спецификация варианта использования «Авторизоваться в системе»

В процессе анализа функциональных требований были определены различные сценарии использования программы. Для основных сценариев использования были разработаны спецификации, которые описывают последовательность действий, выполняемых пользователем и системой.

1.4 Анализ нефункциональных требований

Нефункциональные требования являются ограничением, которое требуют от системы. Нефункциональные требования играют большую роль в разработке программных средств. Они гарантируют, что разрабатываемое программное обеспечение отвечает потребностям пользователя и способна функционировать, как задумано разработчиком [2].

1.4.1 Unity

Unity – платформа для создания кроссплатформенных приложений. Выбор платформы обусловлен несколькими факторами [3].

Главной особенностью Unity является ее кроссплатформенность. Приложение для конструирования и выполнения практических работ школьного курса химии разрабатывается сразу на две мобильные операционные системы: Android и IOS [4].

Помимо этого, Unity предоставляет большие возможности в плане визуализации и создания графики. Благодаря этому приложение может сделать форму обучения для учеников более привлекательной [5].

Ко всем прочему, Unity поддерживает большое количество сторонних плагинов. В разрабатываемом приложении это играет большую роль, поскольку в программном средстве задействовано множество средств Firebase [6].

1.4.2 Firebase Authentication

Firebase Authentication – это сервис, который предоставляет возможность интеграции средства аутентификации и регистрации. Этот сервис удобен тем, что легко интегрируется в разрабатываемое приложение [7]. Firebase Authentication позволяет осуществлять регистрацию и авторизацию с помощью электронной почты, аккаунта Google, Facebook, Twitter и т. д.

1.4.3 Firebase Realtime Database

Firebase Realtime Database – облачная база данных в реальном времени, которая предоставляется платформой Firebase. Это БД позволяет разработчикам создавать отлично масштабируемые приложения с синхронизацией данных [8]. В приложении данная БД хранит информацию о пользователях. Для учеников она хранит информацию о том, какие уроки они прошли и на какие оценки. Для учителей БД хранит информацию о том, какие ученики привязаны к учителю.

1.4.4 Firebase Storage

Firebase Storage – это облачное хранилище, предоставляемое Google как часть платформы Firebase. Данное хранилище предназначено для хранения разных медиафайлов, которые можно удобно скачивать и легко скачивать [9]. В приложении для конструирования и выполнения лабораторных работ по химии

Firestore Storage хранит файлы, которые содержат описание урока, необходимые реагенты для успешного выполнения работы, а также элементы, которые даются ученику. Содержатся эти файлы в формате JSON.

1.4.5 ARCore

ARCore – это платформа дополненной реальности, которую разработала компания Google. Данная платформа предоставляет инструменты для создания приложений с дополненной реальностью [10]. Эти инструменты помогают определять поверхности, что помогает при конструировании практических работ школьного курса химии. Помимо этого, платформа ARCore удобно интегрируется в платформу Unity. Благодаря этому, платформа интегрируется не только для мобильной операционной системы Android, но и iOS.

1.5 Планирование разработки и оценка бюджета

Бюджет проекта является планом всех расходов, необходимые для реализации проекта [11]. Для оценки временных затрат, а также наглядного планирования процесса разработки была построена диаграмма Ганта с использованием инструментов Microsoft Office Project 2007 [12]. Общий процесс разработки приложения содержит четыре вехи. Каждый этап включает в себя некоторый набор подзадач, которые необходимо выполнить для перехода на новый этап. Для разработки данного приложения целесообразно выделить следующие этапы:

- 1) анализ предметной области и постановка задачи на разработку мобильного приложения;
- 2) проектирование мобильного приложения для конструирования практических работ по химии;
- 3) реализация мобильного приложения для конструирования практических работ по химии;
- 4) контроль качества программного обеспечения.

Краткое описание задач каждого этапа приведено на рисунке 9.

	Название задачи	Длительность	Начало	Окончание	Предшественники	Названия ресурсов
1	Начало проекта	0 дней	Пн 08.04.24	Пн 08.04.24		
2	Анализ требований	13 дней	Пн 08.04.24	Ср 24.04.24		
3	Анализ предметной	3 дней	Пн 08.04.24	Пн 15.04.24	1	Аналитик;Ноутбук[1 шт.];Интернет[100,00 Р]
4	Анализ программног	1 день	Вт 16.04.24	Пн 22.04.24	3	Аналитик;Ноутбук[1 шт.];Интернет[50,00 Р]
5	Анализ функциональ	1 день	Вт 23.04.24	Вт 23.04.24	4	Ноутбук[1 шт.];Интернет[50,00 Р];Аналитик
6	Анализ нефункциона	1 день	Ср 24.04.24	Ср 24.04.24	5	Аналитик;Ноутбук[1 шт.];Интернет
7	Анализ требований :	0 дней	Ср 24.04.24	Ср 24.04.24	6	
8	Проектирование моби	9 дней	Чт 25.04.24	Вт 07.05.24		
9	Проектирование арх	2 дней	Чт 25.04.24	Пн 29.04.24	7	Аналитик;Ноутбук[1 шт.];Интернет
10	Проектирование стр	2 дней	Вт 30.04.24	Пн 06.05.24	9	Интернет[50,00 Р];Ноутбук[1 шт.];Аналитик[5
11	Проектирование пол	1 день	Вт 07.05.24	Вт 07.05.24	10	Ноутбук[1 шт.];Разработчик;Интернет
12	Прототипирование и	0 дней	Вт 07.05.24	Вт 07.05.24	9;10;11	
13	Реализация приложен	14 дней	Ср 08.05.24	Пн 27.05.24		
14	Выбор реализации пл	5 дней	Ср 08.05.24	Вт 14.05.24	12	Ноутбук[1 шт.];Разработчик;Интернет
15	Реализация пользо	9 дней	Ср 15.05.24	Пн 27.05.24	14	Ноутбук[1 шт.];Разработчик
16	Реализация приложе	0 дней	Пн 27.05.24	Пн 27.05.24	15	
17	Контроль качества прс	5 дней	Вт 28.05.24	Пн 03.06.24	14	Тестировщик[50%];Ноутбук[1 шт.]
18	Отладка и тестиров	1 день	Вт 28.05.24	Вт 28.05.24	16	Тестировщик;Ноутбук[1 шт.]
19	Модульные тесты	2 дней	Ср 29.05.24	Чт 30.05.24	18	Тестировщик;Ноутбук[1 шт.]
20	Метрики кода	2 дней	Пт 31.05.24	Пн 03.06.24	14;19	Тестировщик;Ноутбук[1 шт.]
21	Реализация приложе	0 дней	Пн 03.06.24	Пн 03.06.24	20	
22	Конец проекта	0 дней	Пн 03.06.24	Пн 03.06.24	21	

Рисунок 9 – Сведения об этапах разработки приложения

Для наглядного отражения временных характеристик разработки программных средств была построена диаграмма Ганта. Данная диаграмма приведена на рисунке 10.

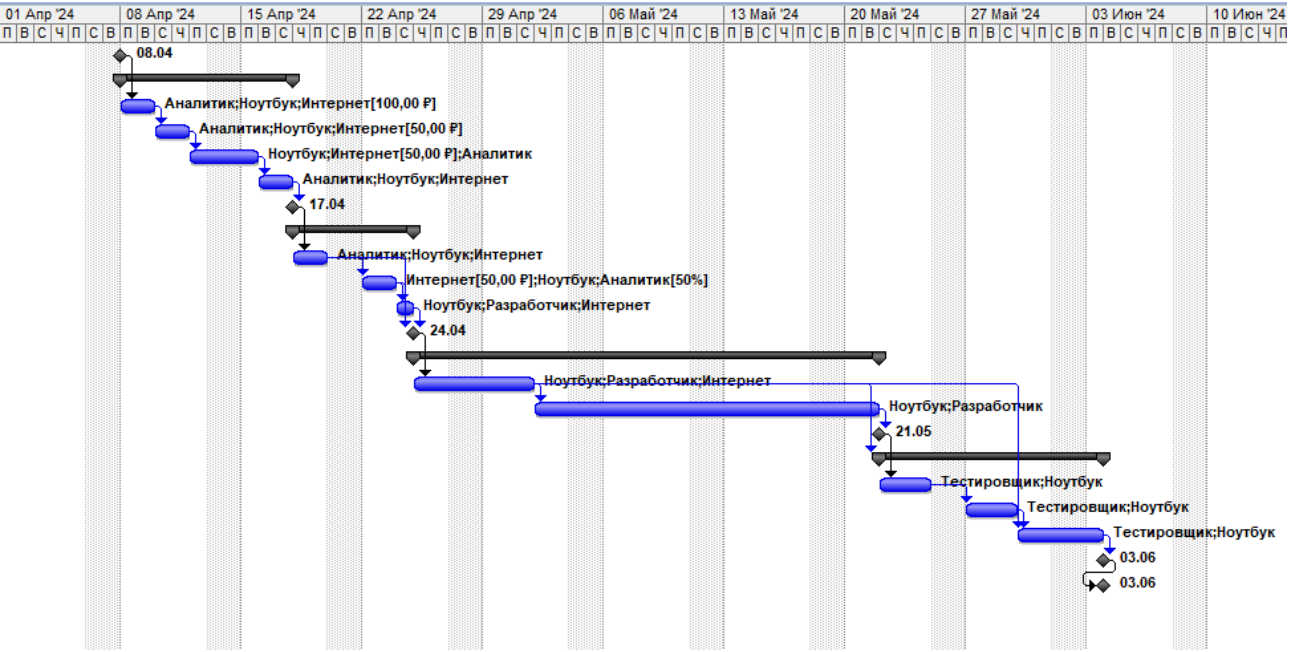


Рисунок 10 – Этапы разработки приложения на диаграмме Ганта

На первом этапе содержатся задачи, направленные на проведение анализа предметной области и формирования исходных требований к разработке программных средств. Второй этап предполагает проектирование программных средств, разработку необходимых графических схем и планирование способов реализации приложения. На третьем этапе предполагается конечная реализация разрабатываемых программных средств, а также выбор способов развертывания

и исполнения. Четвертый этап является заключительным и предполагает выбор и обоснование методов тестирования реализованных программных средств. Итоговый вариант приложения просматривается на наличие ошибок в работе по ряду тестовых случаев.

Для оценки финансовых затрат необходимо назначить цены ресурсов. Стоимость используемых ресурсов представлена на рисунке 11.

	Название ресурса	Стандартная ставка
	Аналитик	110,00 Р/ч
	Разработчик	200,00 Р/ч
	Тестировщик	125,00 Р/мес
	Интернет	
	Ноутбук	100,00 Р/ч

Рисунок 11 – Стоимость задействованных ресурсов

Общая оценка финансовых затрат проекта представлена на рисунке 12.

Название задачи	Затраты
Начало проекта	0,00 Р
☒ Анализ требований	13 640,00 Р
Анализ предметной области	3 460,00 Р
Анализ программного обеспечения	3 410,00 Р
Анализ функциональных требований	3 410,00 Р
Анализ нефункциональных требований	3 360,00 Р
Анализ требований завершен	0,00 Р
☒ Проектирование мобильного приложения	8 290,00 Р
Проектирование архитектуры приложения	3 360,00 Р
Проектирование структуры приложения	2 530,00 Р
Проектирование пользовательского интерфейса	2 400,00 Р
Прототипирование интерфейса завершенно	0,00 Р
☒ Реализация приложения	45 600,00 Р
Выбор реализации приложения	12 000,00 Р
Реализация пользовательского интерфейса	33 600,00 Р
Реализация приложения завершенно	0,00 Р
☒ Контроль качества программного обеспечения	14 484,38 Р
Отладка и тестирование	2 418,75 Р
Модульные тесты	2 418,75 Р
Метрики кода	2 418,75 Р
Реализация приложения завершенно	0,00 Р
Конец проекта	0,00 Р

Рисунок 12 – Общая оценка финансовых затрат

Таким образом, исходя из оценки финансовых и временных затрат, видно, что предполагаемые затраты на разработку составляют 82014,38 рублей, а общая продолжительность – 41 день.

2 Проектирование мобильного приложения выполнения школьных лабораторных работ по химии

2.1 Архитектура приложения

Монолит является наиболее удобной архитектурой для поддержания приложения одним разработчиком [13]. В разрабатываемом мобильном приложении было принято использовать архитектуру «монолит», схема которой приведена на рисунке 13.

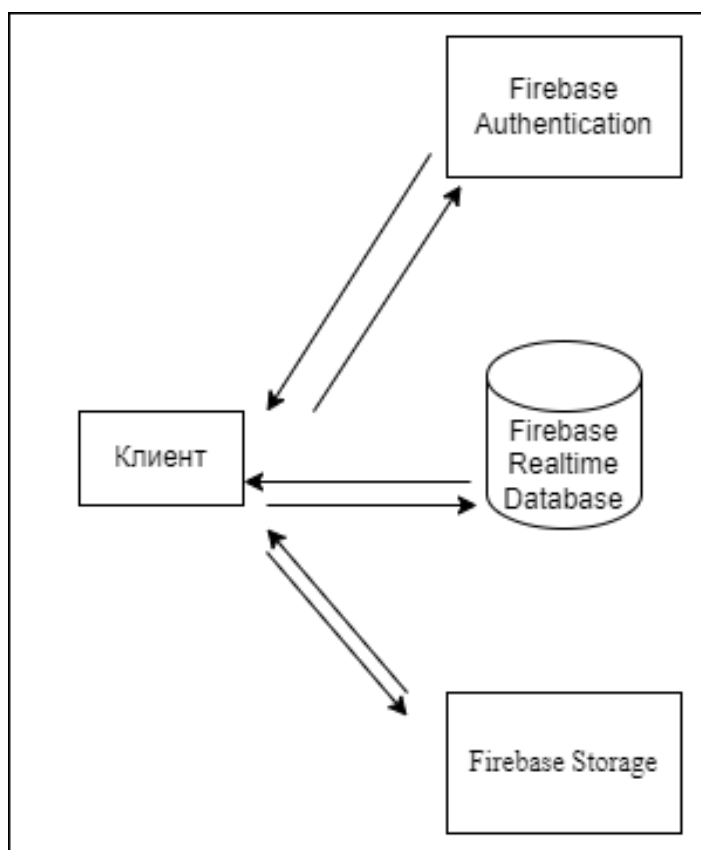


Рисунок 13 – Архитектура «монолит»

В приложении выступают следующие отделы Firebase Authentication, Firebase Realtime Database и Firebase Storage. Первый отдел монолита отвечает за регистрацию новых пользователей, и их вход в приложение. Второй отдел служит в качестве базы данных, которая содержит информацию о пользователях. Последний отдел монолита хранит заготовленные уроки для приложения.

2.2 Проектирование структуры приложения

2.2.1 Концептуальная модель базы данных

Концептуальная модель данных – это наглядное представление структуры данных на абстрактном уровне, позволяющее описать сущности, их характеристики и взаимосвязи без упоминания конкретных технических реализаций [14].

На основе анализа предметной области была составлена диаграмма типа «сущность-связь», представленной на рисунке 14.

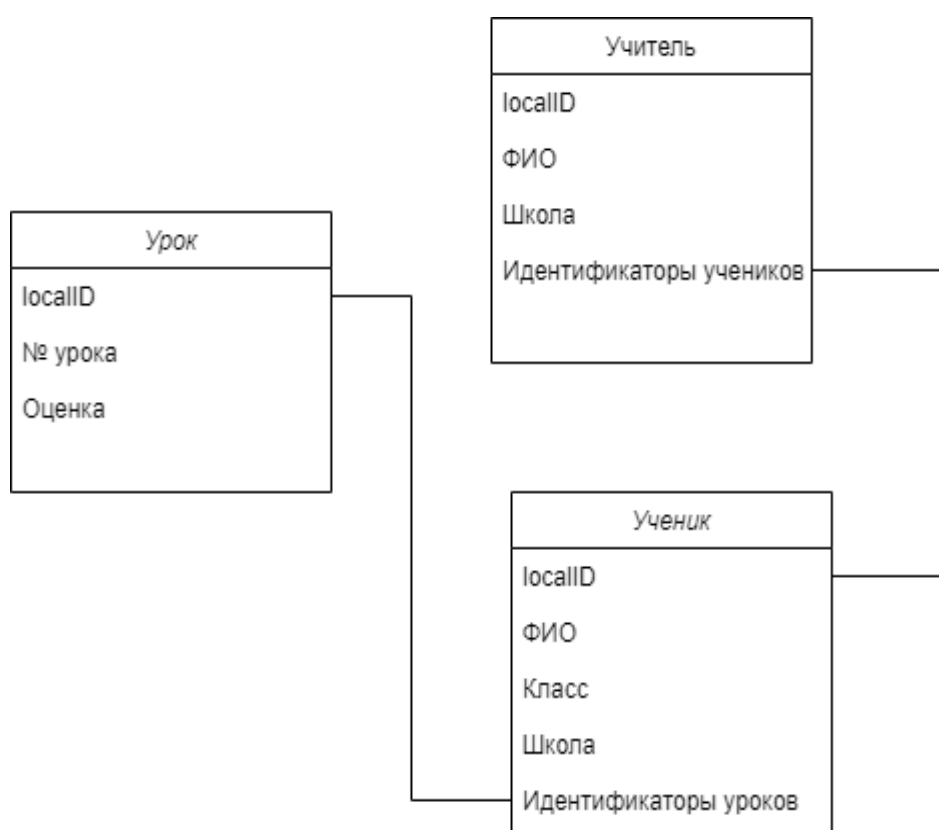


Рисунок 14 – Концептуальная модель базы данных предметной области
«Школьные уроки химии»

На основе концептуальной модели базы данных была составлена таблица сущностей, приведенная в таблице 2.

Таблица 2 – Сущности БД

Название сущности	Идентификатор сущности	Назначение сущности
Учитель	Teacher	Содержит сведения о пользователях приложения с ролью учителя

Продолжение таблицы 2

Название сущности	Идентификатор сущности	Назначение сущности
Ученик	Student	Содержит сведения о пользователях приложения с ролью ученика
Урок	Lesson	Содержит сведения об уроке, проведенным пользователем

Концептуальная модель данных построена в графической среде draw.io в нотации IDEF1X [15].

2.2.2 Логическая модель базы данных

Логическая модель – это расширение концептуальной модели данных. Она включает в себя все сущности, атрибуты, ключи и взаимосвязи, которые представляют бизнес-информацию и определяют бизнес-правила [16].

На основании концептуальной модели была сконструирована логическая модель базы данных, изображенная на рисунке 15.

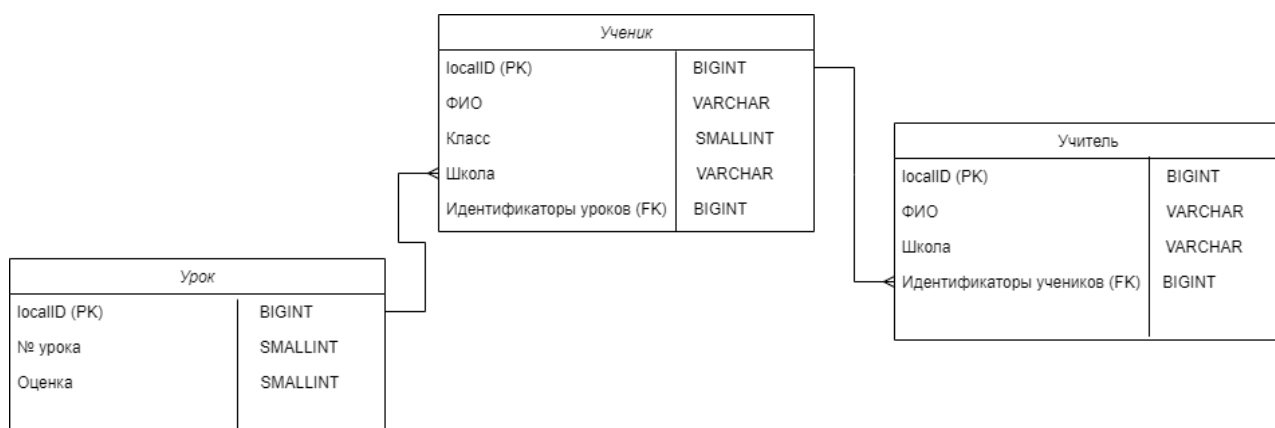


Рисунок 15 – Графическое представление логической модели

Следовательно, созданные визуальные схемы для организации и хранения данных обеспечивают возможность технической реализации доступа к базе данных приложения и наглядного отображения взаимосвязей между данными.

2.3 Проектирование пользовательского интерфейса

Интерфейс мобильного приложения был разработан на основе выявленных функциональных требований. Он включает страницы входа, регистрации, список уроков, информация о них и главный экран, где делается основная часть урока.

Когда вы нажимаете на кнопку «Вход», система осуществляет проверку наличия пользователя с указанным логином и паролем. Если пользователь не обнаружен, появляется сообщение о неверном логине или пароле. В случае успешной авторизации система считает процесс завершенным. Макет страницы входа продемонстрирован на рисунке 16.

The diagram shows a rectangular frame representing a login page. Inside the frame, there are four elements: two empty rectangular input fields at the top, and two buttons at the bottom. The top button is labeled 'Войти' (Login) and the bottom button is labeled 'Регистрация' (Registration). Four lines with numbers 1 through 4 point to these elements from the right side of the frame: line 1 points to the top input field, line 2 points to the bottom input field, line 3 points to the 'Войти' button, and line 4 points to the 'Регистрация' button.

Рисунок 16 – Макет страницы входа

Ниже представлены пояснения к рисунку 16.

- 1 – поле для ввода почты (логина) пользователя;
- 2 – поле для ввода пароля пользователя;
- 3 – кнопка «Вход»;
- 4 – кнопка для перехода на страницу регистрации.

При нажатии кнопки «Зарегистрироваться» система проверяет, являются ли введенные логин и пароль допустимыми. Если нет, будет выведено сообщение. Иначе регистрация будет считаться успешной и будет проведена автоматическая авторизация. Макет страницы регистрации продемонстрирован на рисунке 17.

Макет страницы регистрации, состоящий из пяти горизонтальных текстовых полей ввода, расположенных вертикально друг над другом. Каждое поле имеет номер от 1 до 5, указанный справа с помощью горизонтальной линии. В нижней части макета находится кнопка с текстом «Зарегистрироваться», к которой также проведена линия с номером 6.

Рисунок 17 – Макет страницы регистрации

Макет страницы регистрации содержит следующие элементы:

- 1 – поле ввода электронной почты;
- 2 – поле ввода имени пользователя;
- 3 – поле для ввода названия школы пользователя;
- 4 – поле для ввода класса пользователя;
- 5 – поле для ввода пароля пользователя;

6 – кнопка «Зарегистрироваться».

На странице переходов у ученика есть возможность перейти на необходимую страницу, а именно страница со списком уроков, или страница с информацией о прогрессе ученика. Макет страницы перехода продемонстрирован на рисунке 18.



Рисунок 18 – Макет страницы перехода

Макет страницы перехода к странице возможностями ученика содержит следующие элементы

1 – кнопка для перехода к странице с выбором уроков;

2 – кнопка для перехода к странице с просмотром информации ученика.

Ученик может просматривать, какие уроки он прошел, насколько хорошо и какие уроки предстоит пройти. Макет страницы информации ученика продемонстрирован на рисунке 19.

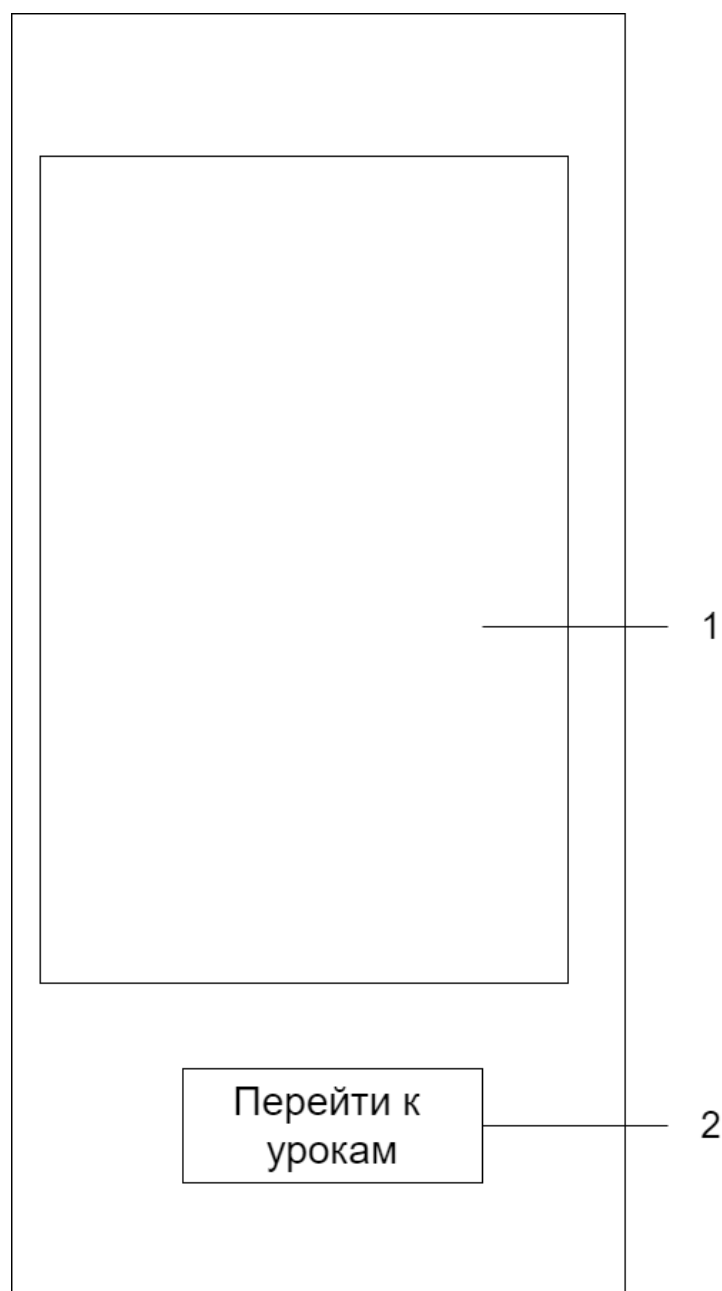


Рисунок 19 – Макет страницы информации ученика

Макет страницы перехода к странице возможностями ученика содержит следующие элементы

- 1 – информация о прохождении уроков приложения;
- 2 – кнопка для перехода к предыдущей странице.

Ученик переходит к выбору действия, которые есть на рисунке 21. Макет интерфейса выбора действий ученика продемонстрирован на рисунке 20.



Рисунок 20 – Макет интерфейса выбора действий

Макет страницы выбора действий содержит следующие элементы:

- 1 – информация с заданием на урок;
- 2 – кнопка для перехода к выбору действий.

Ученик может выбирать объект, который он хочет расположить: химический элемент или колба, в которой можно смешивать или добавлять элементы. Помимо этого, ученику предоставлена возможность выбирать

параметры элементов, а именно вписывать индекс и коэффициент. Макет интерфейса установки объектов продемонстрирован на рисунке 21.

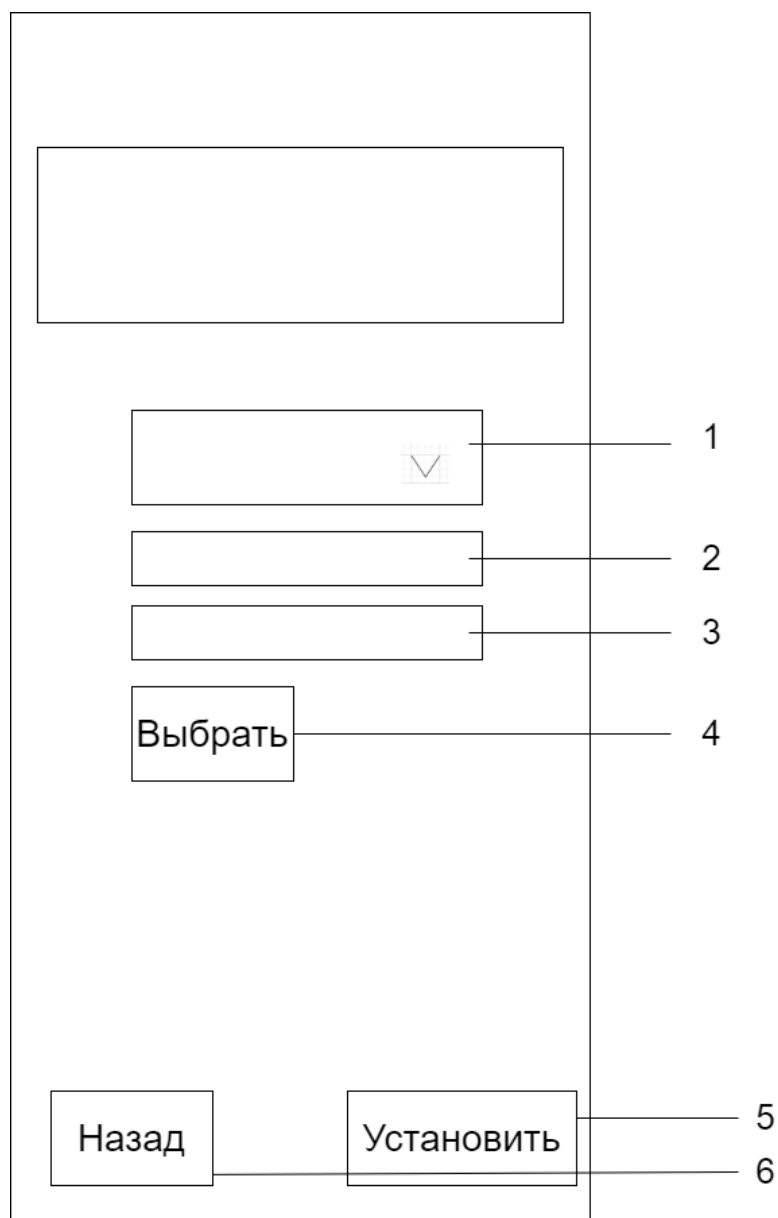


Рисунок 21 – Макет интерфейса установки объектов

Макет этого окна с установкой объектов содержит следующие элементы:

- 1 – выпадающий список с выбором объектов;
- 2 – поле для ввода индекса элемента;
- 3 – поле для ввода коэффициента элемента;
- 4 – кнопка для подтверждения введенных данных;
- 5 – кнопка для перехода к предыдущему окну;
- 6 – кнопка для установки объекта.

Ученик нажимает на объект и выбирает, какое действие с ним сделать: удалить со стола объект или же выбрать взаимодействие с этим объектом. Под взаимодействием подразумевается для элементов – добавить их в реагент, для реагентов – добавить их в основную колбу, в которой производится основное смешение, для колбы – взаимодействие добавленных реагентов. Макет окна с выбором установленных объектов продемонстрирован на рисунке 22.

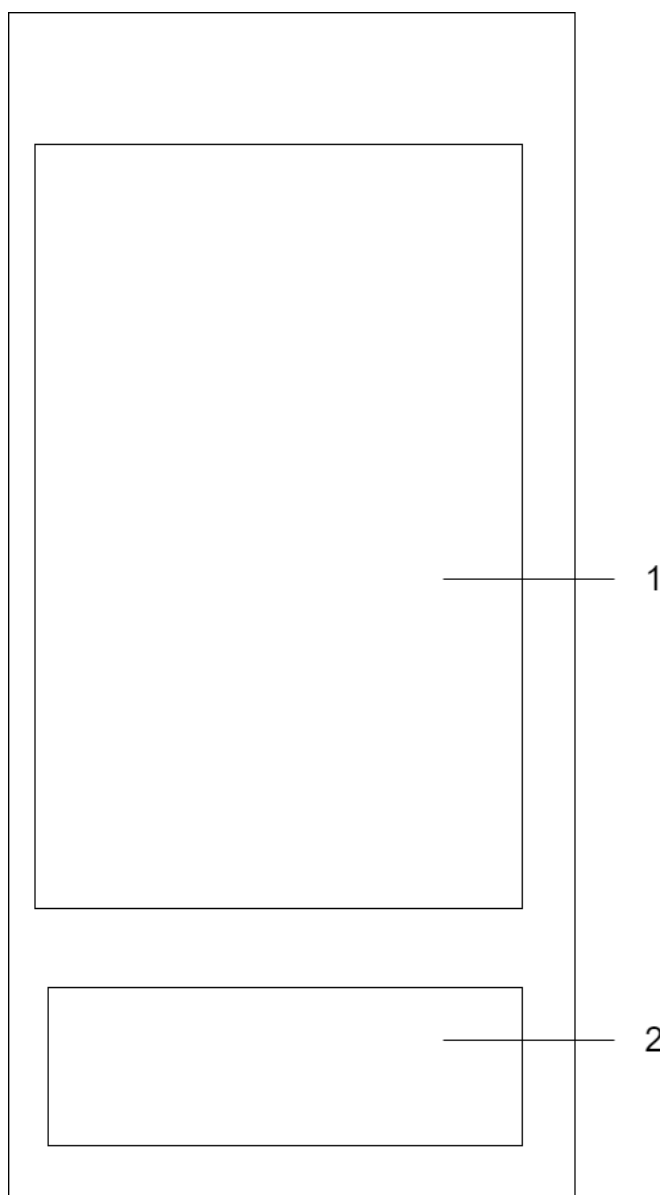


Рисунок 22 – Макет окна с выбором установленных объектов

Макет окна с выбором установленных объектов, представленный на рисунке 22, содержит следующие элементы:

1 – выбираемый объект;

2 – название объекта и действия, которые можно выполнять.

Учитель при входе в аккаунт переходит на страницу, где может выбрать ученика и посмотреть информацию о его прогрессе в приложении. Информация об учениках продемонстрирована на рисунке 19. Макет страницы со списком учеников продемонстрирован на рисунке 23.

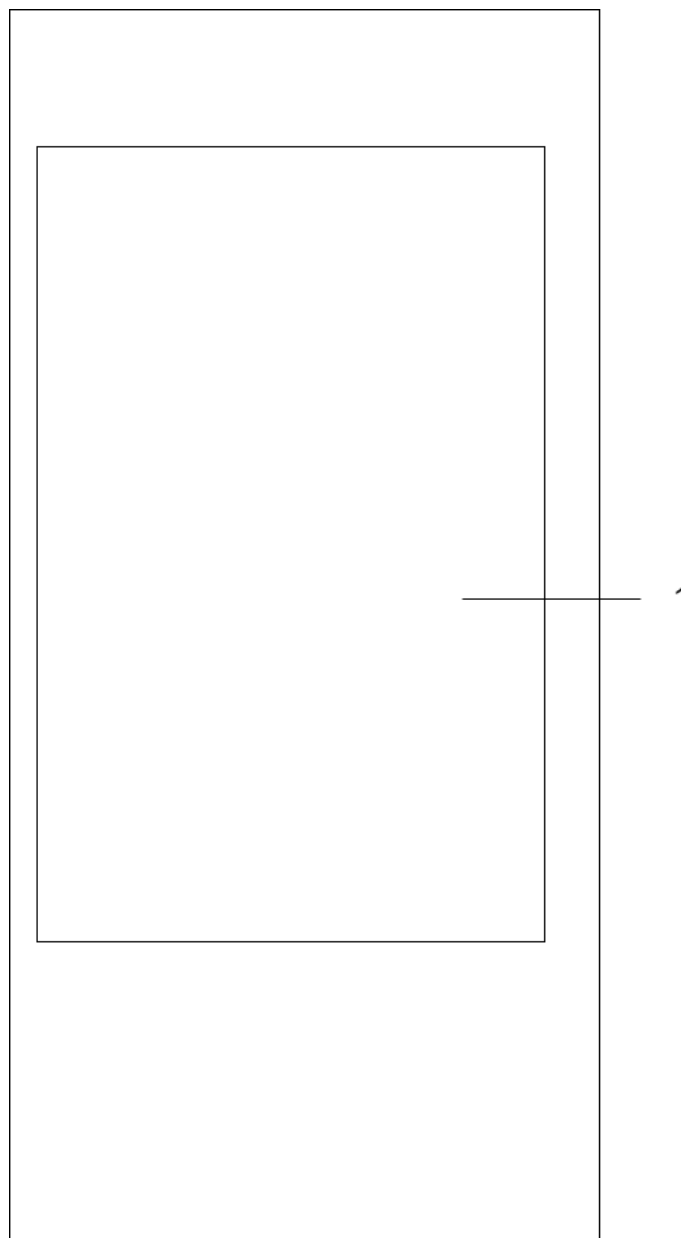


Рисунок 23 – Макет страницы со списком учеников

На экране появляется список уроков. Ученик может выбрать, какой урок пройти. Макет страницы со списком уроков продемонстрирован на рисунке 24.

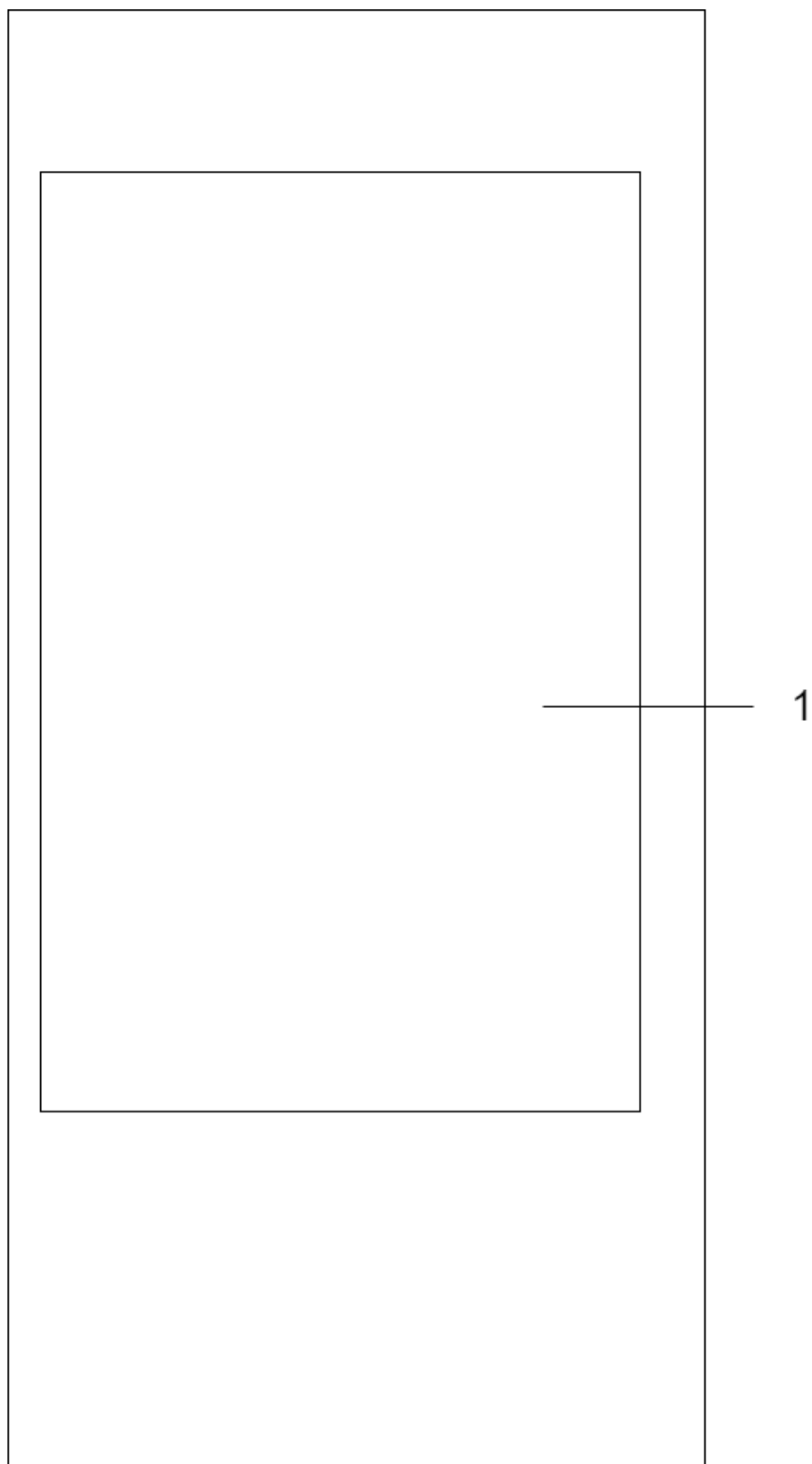


Рисунок 24 – Макет Страницы выбора урока

Информация, что необходимо сделать в этом уроке. При нажатии на кнопку «Приступить к уроку» будет переход к основному меню.

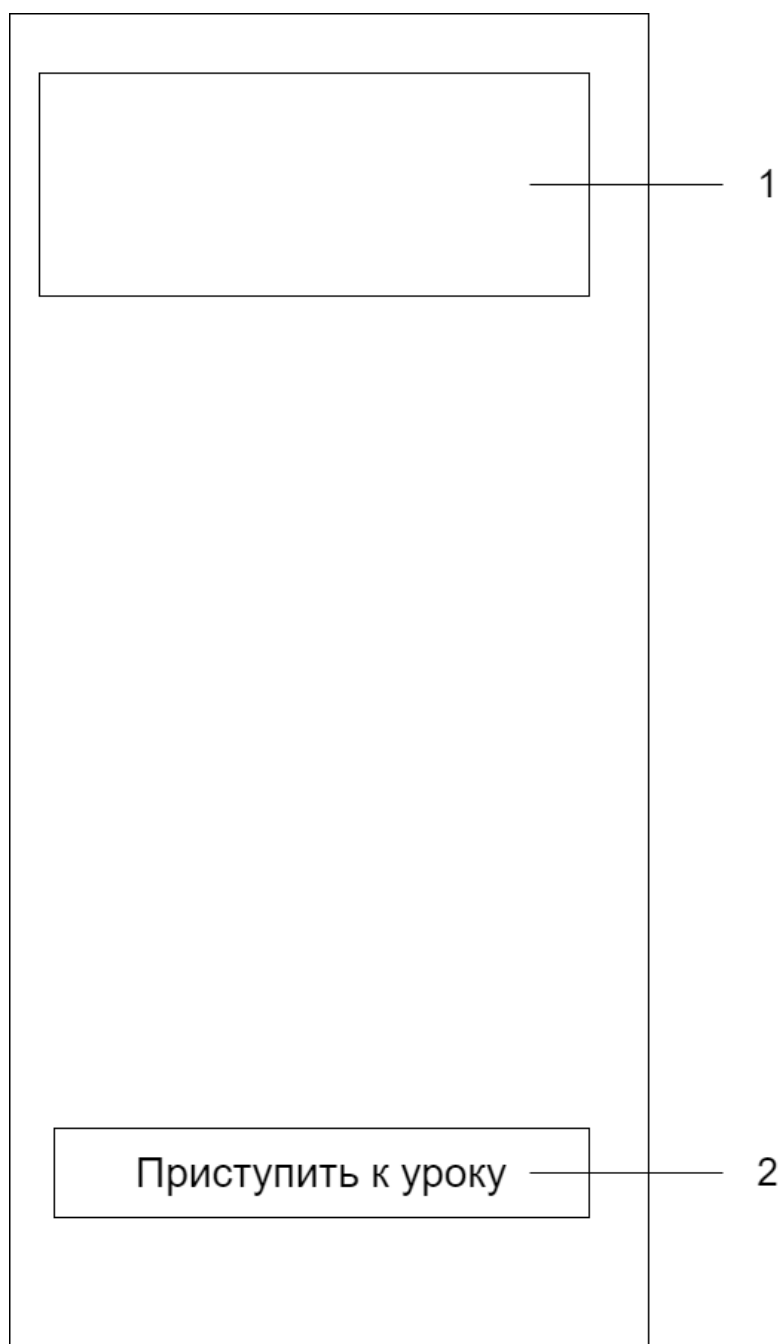


Рисунок 25 – Макет страницы информации об уроке

Макет страницы информации об уроке, приведенный на рисунке 25, имеет следующие элементы:

- 1 – поле для отображения задачи выбранного урока;
- 2 – кнопка для начала урока.

Интерфейс мобильного приложения спроектирован с учетом функциональных требований.

3 Реализация приложения для проведения школьных практических работ по химии

3.1 Реализация базы данных

Физическая модель базы данных представляет собой логическую модель. В рамках данного приложения используется БД Firebase Realtime Database.

Физические модели баз данных определяют способы размещения данных в среде хранения и способы доступа к этим данным, которые поддерживаются на физическом уровне.

Согласно концептуальной и логической модели, был разработана физическая модель данных, представленная на рисунке 26.

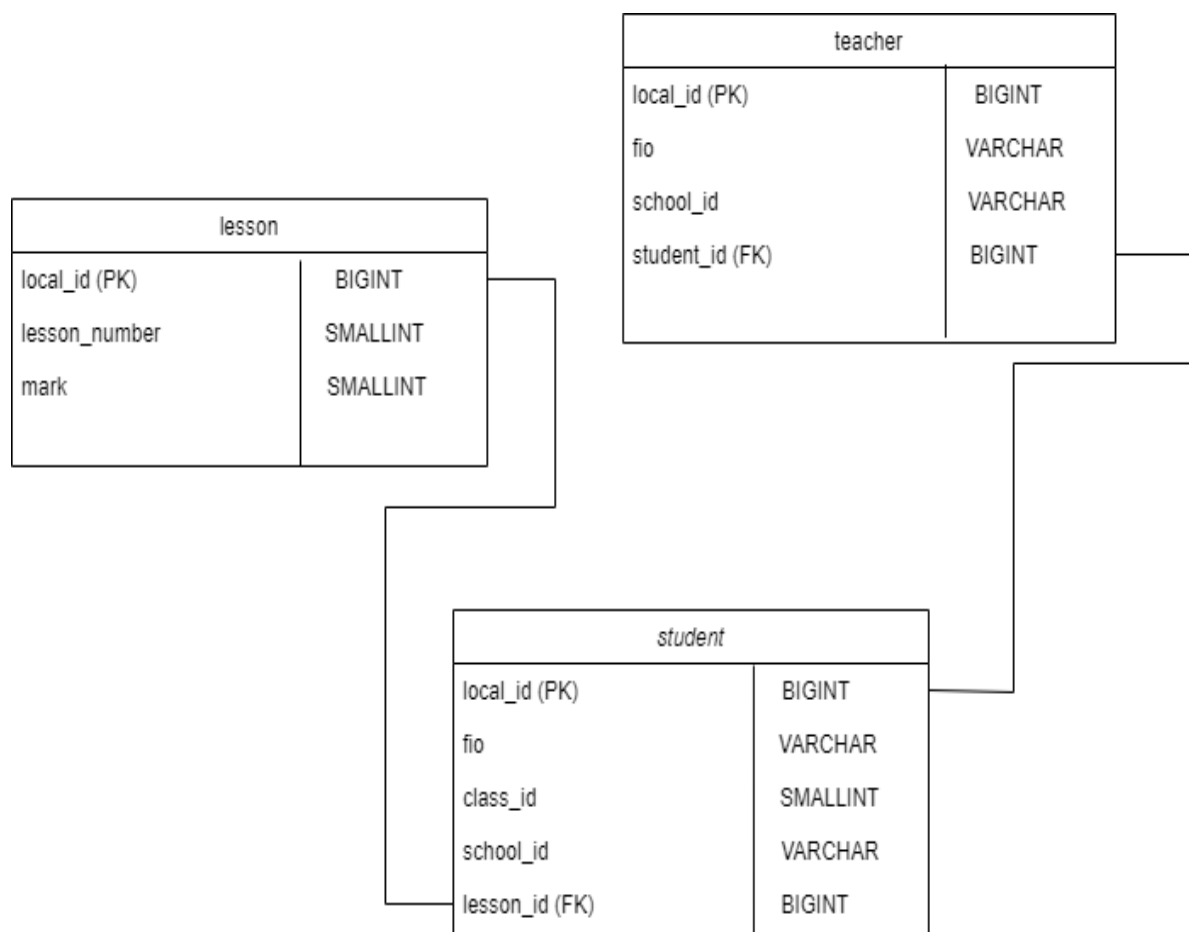


Рисунок 26 – Физическая модель базы данных

На основе физической модели, была разработана база данных с использованием Firebase Realtime Database. В таблице 3 представлена спецификация сущности «Урок».

Таблица 3 – Спецификация сущности «Урок»

Наименование	Описание	Тип данных	Возможность значения NULL	ОДЗ	Значение по умолчанию	Роль
local_id	Идентификатор	BIGINT	Нет	> 0	–	PK
lesson_number	Номер урока	SMALLINT	Да	>= 0	–	–
mark	Оценка	SMALLINT	Да	>= 0	–	–

В таблице 4 представлена спецификация сущности «Учитель».

Таблица 4 – Спецификация сущности «Учитель»

Наименование	Описание	Тип данных	Возможность значения NULL	ОДЗ	Значение по умолчанию	Роль
local_id	Идентификатор	SMALLINT	Нет	> 0	–	PK
fio	Полное имя	VARCHAR	Нет	Любое	–	–
school_id	Номер школы	VARCHAR	Нет	Любое	–	–
student_id	Идентификатор ученика	VARCHAR	Да	Любое	–	FK

В таблице 5 представлена спецификация сущности «Ученик».

Таблица 5 – Спецификация сущности «Ученик»

Наименование	Описание	Тип данных	Возможность значения NULL	ОДЗ	Значение по умолчанию	Роль
local_id	Идентификатор	BIGINT	Нет	> 0	–	PK
fio	Полное имя	VARCHAR	Нет	Любое	–	–
class_id	Идентификатор класса	SMALLINT	Нет	> 0	–	–
school_id	Номер школы	VARCHAR	Нет	Любое	–	–
lesson_id	Идентификатор урока	BIGINT	Нет	> 0	–	FK

Таким образом, реализованная физическая схема базы данных позволяет эффективно хранить сведения о прогрессе пользователей в приложении.

3.2 Реализация пользовательского интерфейса

В соответствии с макетами интерфейса, представленными в пункте 2.3, были реализованы страницы мобильного приложения. Результаты реализации продемонстрированы в приложении Б.

С помощью Unity можно создавать приложения для операционных систем Android, iOS и Windows на основе общей базы кода. Unity позволяет создавать графический интерфейс с помощью визуальных средств, которые можно удобно передвигать. Помимо этого, с помощью кода на языке программирования C# создавались анимации.

3.3 Реализация состояния переходов в приложении

При выполнении практического урока в приложении ученик часто взаимодействует с объектом колба с реагентом, которая часто меняет свои состояния. Диаграмма состояний переходов колбы с реагентом представлена на рисунке 27



Рисунок 27 – Диаграмма состояний колбы с реагентом

Более подробное описание состояний колбы с химикатами описано в таблице 6.

Таблица 6 – Описание состояний колбы

Состояния	Описание состояния
Выбрана	Из списка химикатов выбирается нужный и формируется колба с этим химикатом
Перемещена	После выбора химиката, можно перемещать колбу с химикатом, чтобы разместить её.
Размещена	Размещение колбы с выбранным химикатом
Удалена	Удаление колбы с химикатом
Использована	Колба с химикатом использована для смешения с другими химикатами

Таблица 6 отражает все состояния колбы, в связи с чем легче понять, как у ученика есть возможности взаимодействия с ней.

3.4 Реализация программных средств

Для приложения была спроектирована диаграмма классов, приведенная на рисунке 28. Данная диаграмма отражает основные технические решения, применяемые в рамках разработки приложения.

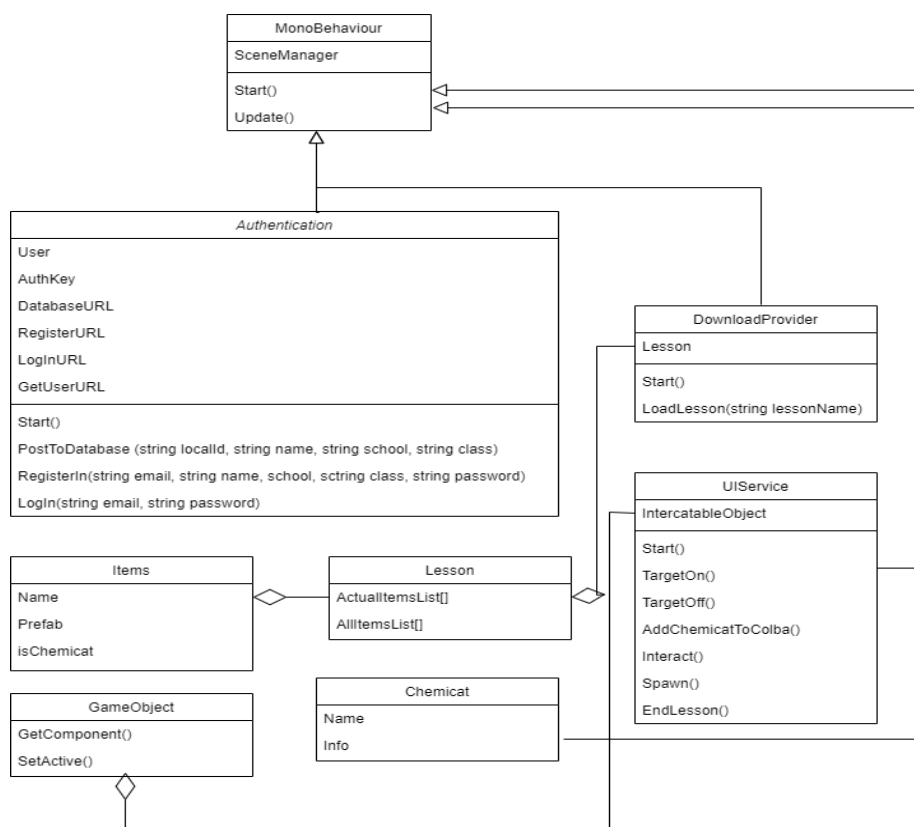


Рисунок 28 – Диаграмма классов

В таблице из приложения В представлена спецификация классов основного приложения.

4 Контроль качества программного обеспечения

4.1 Тестирование черным ящиком

Тестирование черным ящиком подразумевает функциональное и нефункциональное тестирование без доступа к внутренней системе компонентов [17].

4.1.1 Классы эквивалентности

Логин должен содержать 5 и более символов: цифры, специальные символы, латинские буквы. Логин должен заканчиваться знаком «@» и латинскими буквами

Таблица 7 – Классы эквивалентности

	Правильные КЭ	Неправильные КЭ
Длина	6 – 35 (1)	0 – 5 (2) 35 – ∞ (3)
Структура	Регулярное выражение: $^[A - Za - z0 - 9.\%+-]+\@[A - Za - z0 - 9.-]+\.[A - Za - z]{2,}\$$ (4)	Остальное (5)
Содержание	Буквы Латинские (6) Цифры (7) Спецсимволы (8)	Остальное (9)

Графическое представление необходимой длины логина представлено на рисунке 29.

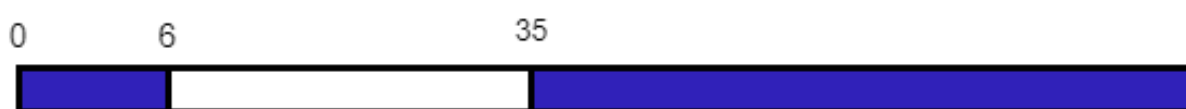


Рисунок 29 – Классы эквивалентности: длина

Графическое представление содержания продемонстрировано на рисунке 30.

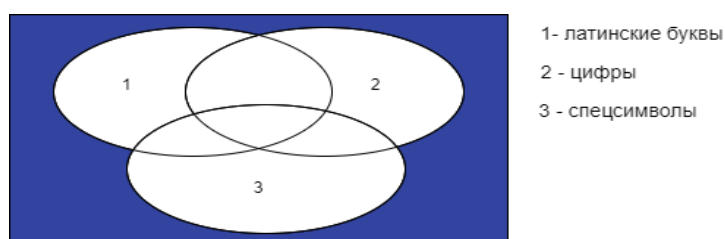


Рисунок 30 – Классы эквивалентности: содержание

Тест кейсы для логина отражены в таблице 9.

Таблица 8 – Тест–кейсы

№	КЭ	Входные данные	Выходные данные
0	Все правильные	ASfsdf5@gmai.com	Успешно
1	1,4 ,6,7,8 2,4,6,7,8	@1.f	Ошибка: меньше 5 символов
2	3,4,6,7,8	gdrgrGR1wasegegheghergerger@sdfs.fsafrgwefwef	Ошибка: меньше 35 символов
3	1,4,5,6,7	sAm3@g.!	Ошибка: логин должен не может содержать спецсимволы после символа «@»
4	1,4,6,7	Som3@fam	Ошибка: логин должен содержать точку после символа «@»
5	1,4,5,7	argfadf	Ошибка: логин должен содержать символ «@»
6	1,4,5,6,7	Arg3@.com	Ошибка: логин должен содержать букву после символа «@» и перед точкой

Все тест-кейсы, указанные в таблице 9, успешно пройдены.

4.1.2 Граничные условия

Выделим граничные условия для логина пользователя, что можно увидеть ниже в таблице 9.

Таблица 9 – Граничные условия

Границы	Условия
Физические	Логин неотрицательной длины
Логические	Логин содержит больше 1 символа

Продолжение таблицы 9

Границы	Условия
Технологические	Логин не может быть длиной больше максимально возможной (35)
Произвольные	См. 1.1. Классы эквивалентности

Тест-кейсы для граничных условий отражены в таблице 10

Таблица 10 – Тест-кейсы для граничных условий

№	Границы	Входные данные	Выходные данные
7	Ф	some@gmail.com	Успешно
8	Л	T9@	Ошибка: меньше 6 символов
9	Т	Wq#25abcD5sdfsqwef	Ошибка: больше 35 символов

Тест кейсы, указанные в таблице 10, успешно пройдены.

4.2 Модульные тесты

Модульное тестирование, иногда блочное тестирование или юнит-тестирование (англ. unit testing) – процесс в программировании, позволяющий проверить на корректность отдельные модули исходного кода программы, наборы из одного или более программных модулей вместе с соответствующими управляющими данными, процедурами использования и обработки [18].

Идея состоит в том, чтобы писать тесты для каждой нетривиальной функции или метода. Это позволяет быстро проверить, не привело ли очередное изменение кода к регрессии, то есть к появлению ошибок в уже оттестированных местах программы, а также облегчает обнаружение и устранение таких ошибок. Например, обновить используемую в проекте библиотеку до актуальной версии можно в любой момент, прогнав тесты и выявив несовместимости.

Выполним разработку модульных тестов алгоритма для тестовых случаев, приведенных в таблице 11.

Таблица 11 – Тестовые случаи для проверки алгоритма оптимизации

№	Содержание	Входные данные	Ожидаемый результат
1	Проверка общей работоспособности регистрации	Логин – ASfsdf5@gmai.com Пароль –password	Успешный регистрация

Продолжение таблицы 11

№	Содержание	Входные данные	Ожидаемый результат
2	Проверка регистрации при длине логина менее 5 символов	Логин – @1.f Пароль – password	Выброшена ошибка регистрации
3	Проверка регистрации при длине логина более 35 символов	Логин – gdrGR1wasegegheghergerger@sdfs.fsafrgwefwef Пароль – password	Выброшена ошибка регистрации
4	Проверка регистрации при длине пароля менее 6 символов	Логин – ASfsdf5@gmai.com Пароль – pass	Выброшена ошибка регистрации
5	Проверка регистрации при некорректной структуре логина	Логин – ASfsdf5@.com Пароль – pass	Выброшена ошибка регистрации

Код тестовых методов описан в листинге 1.

Листинг 1 – Код тестовых методов

```
using System.Collections;
using System.Collections.Generic;
using NUnit.Framework;
using UnityEngine;
using UnityEngine.TestTools;

public class UnitTests
{
    // A Test behaves as an ordinary method
    [Test]
    public void TestSucess()
    {
        Authentication auth = new Authentication();
        bool isSuccess = auth.RegisterIn("ASfsdf5@gmai.com",
"some", "erg", "dge", "password");
        Assert.AreEqual(isSucess, true);
    }
    [Test]
    public void ShortLoginTest()
    {
        Authentication auth = new Authentication();
        bool isSuccess = auth.RegisterIn("@1.f", "some", "erg",
"dge", "password");
        Assert.AreEqual(isSucess, false);
    }
    [Test]
    public void LongLoginTest()
    {
        Authentication auth = new Authentication();
```

```

        bool isSuccess =
auth.RegisterIn("gdrgrGRlwasegegheghegerger@sdfs.fsafrgwefwef",
"some", "erg", "dge", "password");
        Assert.AreEqual(isSuccess, false);
    }
    [Test]
    public void ShortPasswordTest()
    {
        Authentication auth = new Authentication();
        bool isSuccess = auth.RegisterIn("ASfsdf5@gmai.com",
"some", "erg", "dge", "pass");
        Assert.AreEqual(isSuccess, false);
    }
    [Test]
    public void WrongStructureLoginTest()
    {
        Authentication auth = new Authentication();
        bool isSuccess = auth.RegisterIn("ASfsdf5@.com", "some",
"erg", "dge", "password");
        Assert.AreEqual(isSuccess, false);
    }
}

```

Результаты исполнения модульных тестов приведены на рисунке 31.

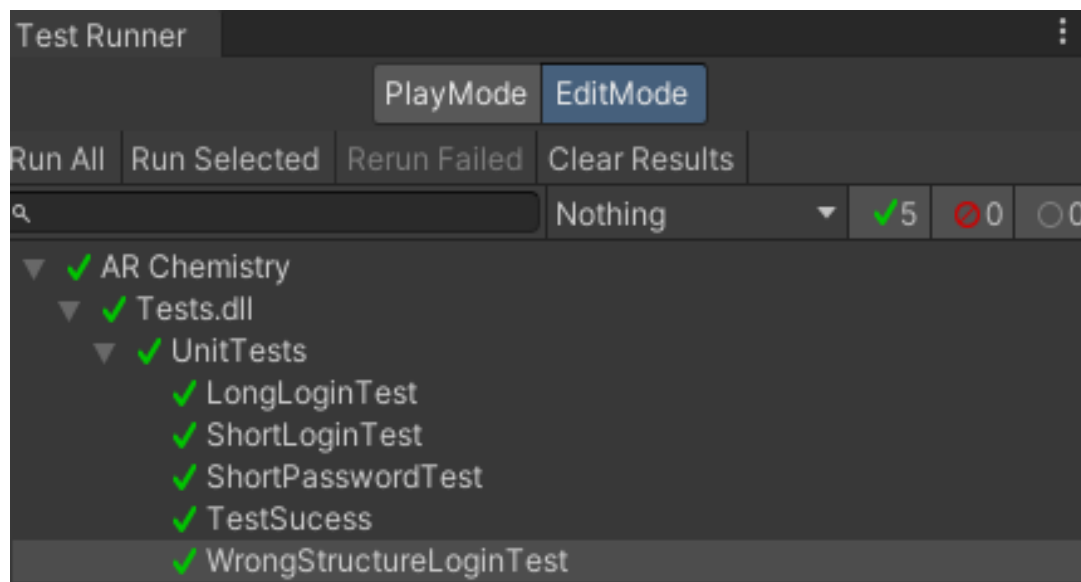


Рисунок 31 – Результат выполнения модульных тестов в Unity

Все модульные тесты пройдены успешно, в связи с чем программа работает корректно.

4.3 Метрики кода

4.3.1 Метрика SLOC

Количество строк кода (англ. Source Lines of Code – SLOC) – это метрика программного обеспечения, используемая для измерения его объема с помощью подсчета количества строк в тексте исходного кода. Функция приведена в приложении [19].

В данной программе имеем:

- 121 физических строк кода;
- 43 логических строк кода;
- строк комментариев.

4.3.2 Метрика Холстеда

Метрика Холстеда относится к метрикам, вычисляемым на основании анализа числа строк и синтаксических элементов исходного кода программы [20].

Основу метрики Холстеда составляют четыре измеряемые характеристики программы:

- NUOprtr (Number of Unique Operators) – число уникальных операторов программы, включая символы-разделители, имена процедур и знаки операций (словарь операторов);
- NUOprnd (Number of Unique Operands) – число уникальных операндов программы (словарь операндов);
- Noprtr (Number of Operators) – общее число операторов в программе;
- Noprnd (Number of Operands) – общее число операндов в программе.

Операторы приведены ниже в таблице 12.

Таблица 12 – Операторы

Операторы	Число вхождений
=	10
==	2
.	59
new	1

Продолжение таблицы 12

Операторы	Число вхождений
If	5
return	3
Start	1
SelectItemOn	1
TargetOn	1
TargetOff	2
InteractWithObjectMenu	1
OpenInteractableMenu	2
foreach	1
var	1
{}	24
n1 = 15	N1 = 114

Операнды приведены ниже в таблице 13.

Таблица 13 – Операнды

Операнды	Число вхождений
firstLesson	2
listOfObject	1
_dropdown	2
_optionMenu	2
_selectItemMenu	2
_spawnObjectMenu	2
_objectInfoMenu	1
_chemicatMenu	2
_chemicatName	2
_chemicatInfo	1
_interactableObject	5
_colbaMenu	2
OptionsMenu	1
SelectItemMenu	1
SpawnObjectMenu	1
items	5
dropdown	3
interactableObject	7
n2 = 18	N2 = 42

Длина программы рассчитывается по формуле:

$$N = N_1 + N_2 = 114 + 42 = 156,$$

где N – длина программы;

N_1, N_2 – число вхождений операторов и операндов.

Словарь программы рассчитывается по формуле:

$$n = n_1 + n_2 = 15 + 18 = 33,$$

где n – общее число операторов;

n_1, n_2 – число операторов и операндов.

Объем программы рассчитывается по формуле:

$$HV = N \cdot \log_2 n = 156 \cdot 5.04 = 786.24,$$

где HV – объем программы;

n – число операторов;

N – число операндов.

Потенциальный объем программы:

$$V = (2 + n) \cdot \log_2(2 + n) = 5 \cdot \log_2(5) = 11.6,$$

где V – потенциальный объем программы;

n – число операторов.

Теоретический уровень программы рассчитывается по формуле:

$$L' = \frac{V}{HV} = \frac{11.6}{786.24} = 0.015,$$

где L' – теоретический уровень программы;

HV – объем программы;

V – потенциальный объем программы.

Сложность программы рассчитывается по формуле:

$$HD = \left(\frac{n_1}{2}\right) \cdot \left(\frac{N_2}{n_2}\right) = \frac{15}{2} \cdot \frac{156}{18} = 65,$$

где HD – сложность программы;

n_1, n_2 – число операторов и операндов;

N_2 – число вхождений операндов.

Проведено тестирование, которое показывает отсутствие критических ошибок и работоспособность программы.

Заключение

Результатом выполнения выпускной квалификационной работы является разработанное приложение, позволяющее конструировать и выполнять практические работы по школьному курсу химии 9 класса.

На стадии анализа требований проведен анализ предметной области изучения химии школьного курса 9 класса, сравнительный анализ программного обеспечения аналогичного назначения, а также выявлены функциональные и нефункциональные требования к разрабатываемым программным средствам.

Проведено структурное и календарное планирование, а также выполнена оценка стоимости разработки приложения. По итогу, на выполнение всех этапов разработки проекта необходим 41 день и 82014,38 рублей.

На стадии проектирования разработана концептуальная и логическая модели базы данных, а также спроектирована архитектура программного продукта. Реализованные программные средства функционально соответствуют поставленным задачам и предоставляют возможности для выполнения практических работ по химии 9 класса. Созданное приложение разворачивается мобильных устройствах с операционными системами Android и IOS. Работоспособность приложения проверена модульным тестированием. Все тесты были успешно пройдены.

Таким образом, задача, поставленная по реализации приложения для конструирования и выполнения практических работ школьного курса химии, выполнена в полном объеме.

Список использованных источников

1. Анализ требований // Хабр [Электронный ресурс]. URL: <https://habr.com/ru/articles/340956/> (дата обращения 03.06.2024).
2. Что такое нефункциональные требования: примеры, определение, полное руководство // Решение Visure [Электронный ресурс]. URL: <https://visuresolutions.com/ru/blog/non-functional-requirements/> (дата обращения 03.06.2024).
3. Корнилов, А. В. UNITY. Полное руководство / А. В. Корнилов. – Санкт-Петербург: Наука и техник, 2020. – 432 с.
4. Хокинг, Д. С. Unity в действии. Мультиплатформенная разработка на C# / Д. С. Хокинг. – Питер: Питер, 2019. – 352 с.
5. Ламмерс, Кенни Шейдеры и эффекты в Unity / Кенни Ламмерс. – Москва: ДМК Пресс, 2014. – 274 с.
6. Jiadong, Chen Game Development with Unity for .NET Developers: The ultimate guide to creating games with Unity and Microsoft Game Stack / Chen Jiadong. – Birmingham, England: Packet Publishing, 2022. – 584 с.
7. Аутентификация Firebase // Documentation Firebase Authentication [Электронный ресурс]. URL: <https://firebase.google.com/docs/auth?hl=ru> (дата обращения 03.06.2024).
8. База данных Firebase в реальном времени // Documentation Firebase Realtime Database [Электронный ресурс]. URL: <https://firebase.google.com/docs/database?hl=ru> (дата обращения 03.06.2024).
9. Облачное хранилище для Firebase // Documentation Firebase Storage [Электронный ресурс]. URL: <https://firebase.google.com/docs/storage?hl=ru> (дата обращения 03.06.2024).
10. Обзор ARCore и поддерживаемых сред разработки // Google for Developers [Электронный ресурс]. URL: <https://developers.google.com/ar/develop?hl=ru> (дата обращения 03.06.2024).
11. Планирование бюджета и построение финансового прогноза в проектах// Projecto [Электронный ресурс]. URL:

<https://projecto.pro/blog/theory/planirovanie-byudzheta-i-postroenie-finansovogo-prognoza-v-proektah/> (дата обращения 03.06.2024).

12. Microsoft Office Project 2007 – Microsoft Lifecycle// Microsoft Learn [Электронный ресурс]. URL: <https://learn.microsoft.com/ru-ru/lifecycle/products/microsoft-office-project-2007> (дата обращения 03.06.2024).

13. Ньюмен, С. От монолита к микросервисам // С. Ньюмен. – Санкт-Петербург: БХВ-Петербург, 2021. – 272 с.

14. Концептуальная модель базы данных // Пензенский Государственный Университет Архитектуры и Строительства. [Электронный ресурс]. URL: <https://studfile.net/preview/9959196/page:2/> (дата обращения 03.06.2024).

15. Верников, Г. Основы методологии IDEF1X // Проект «Корпоративный менеджмент» [Электронный ресурс]. URL: <https://www.cfin.ru/vernikov/idef/idef1x.shtml/> (дата обращения 03.06.2024).

16. Базы данных // МФПА. [Электронный ресурс]. URL: <https://infdis.narod.ru/db/db-10.htm> (дата обращения 03.06.2024).

17. Особенности тестирования «черного ящика» // Лаборатория качества [Электронный ресурс]. URL: <https://quality-lab.ru/blog/key-principles-of-black-box-testing/> (дата обращения 03.06.2024).

18. Карпенко, Н. В. Тестирование информационных систем: модульное, интеграционное, системное. // Н. В. Карпенко. – Лира: Лира, 2016. – 283 с.

19. Рыжков, Е. Программный код и его метрики // Тематический блог «Хабр». [Электронный ресурс]. URL: <https://habr.com/ru/companies/intel/articles/106082/> (дата обращения 03.06.2024).

20. Значения метрик кода // Официальная документация Microsoft [Электронный ресурс] URL: <https://learn.microsoft.com/ru-ru/visualstudio/code-quality/code-metrics-values?view=vs-2022> (дата обращения 03.06.2024).

Приложение А
(рекомендуемое)
Листинг приложения

Листинг A.1 – Файл Authentication.cs

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Text.Json;
using DG.Tweening;
using Display;
using Proyecto26;
using FullSerializer;
using Newtonsoft.Json;
using TMPPro;
using UnityEngine;
using UnityEngine.SceneManagement;
using JsonSerializer = System.Text.Json.JsonSerializer;

public class Authentication : MonoBehaviour
{
    private User flag;
    public string resp;
    public GameManager GameManager;

    public string idToken;
    public string localId;
    private string AuthKey =
    "AIzaSyC6YMYTccbhcxuxuY_p3vzIurh7X9Qv7Ko";
    private string databaseURL = "https://interestingchemistry-
    1b2b3-default-rtdb.firebaseio.com/";
    private string RegisterURL =
    "https://identitytoolkit.googleapis.com/v1/accounts:signUp?key=";
    private string LogInURL =
    "https://identitytoolkit.googleapis.com/v1/accounts:signInWithPassword?ke
    y=";
    private string GetUsersURL = "https://interestingchemistry-
    1b2b3-default-rtdb.firebaseio.com/Users.json";

    private fsSerializer _serializer;
    [Header("LogIn")]

    [SerializeField] private GameObject LogInMenu;
    [SerializeField] private TMP_Text LogInEmail;
    [SerializeField] private TMP_Text LogInPassword;
    [SerializeField] private TMP_Text errorTextLogIn;
    [Header("LogUp")]
    [SerializeField] private TMP_Text LogUpEmail;
    [SerializeField] private TMP_Text LogUpName;
    [SerializeField] private TMP_Text LogUpSchool;
    [SerializeField] private TMP_Text LogUpClass;
    [SerializeField] private TMP_Text LogUpPassword;
    [SerializeField] private TMP_Text errorTextLogUp;

    [Header("UserMenu")]
    [SerializeField] private GameObject UserMenu;
    [SerializeField] private GameObject[] Lessons;
    [SerializeField] private Transform Panel;
```

```

public void Start()
{
    _serializer = new fsSerializer();
    GameManager some =
AllServices.Container.Single<GameManager>();
    if (!some.ActualUser.UserName.Equals(""))
    {
        LogIn(some.Email, some.Password);
    }
}

public void RegisterIn(string email, string name, string school,
string Class, string password)
{
    string userData =
"{\"email\":\""+email+"\", \"password\":\""+password+"\", \"returnSecureTok
en\":true}";
    RestClient.Post<SignResponse>(RegisterURL + AuthKey,
userData).Then(
        response =>
        {
            localId = response.localId;
            idToken = response.idToken;
            PostToDataBase(localId, name, school, Class);
        }).Catch(error =>
        {
            errorTextLogUp.gameObject.SetActive(true);
            errorTextLogUp.text = "Ошибка регистрации, проверьте
данные";
        });

    //PostToDataBase(localId, name, school, Class);
}

public void SignUp()
{
    RegisterIn(LogUpEmail.text, LogUpName.text,
LogUpSchool.text, LogUpClass.text, LogUpPassword.text);
}
public void SignIn()
{
    LogIn(LogInEmail.text, LogInPassword.text);
}

public void PostToDataBase(string localId, string Name, string
school, string Class)
{
    User user = new User(Name, localId, school, Class );
    user.Lessons[0] = true;
    RestClient.Put(databaseURL+"Users/"+localId+".json",
user);
}

public void DoSome()
{
    fsData userData = fsJsonParser.Parse(resp);

```

```

Dictionary<string, User> users = null;
_serializer.TryDeserialize(userData, ref users);
foreach (var user in users.Values)
{
    Debug.Log(user.LocalId);
    if (user.LocalId.Equals(localId))
    {
        GameManager.ActualUser = new User();
        GameManager.ActualUser.UserName = user.UserName;
        GameManager.ActualUser.Class = user.Class;
        GameManager.ActualUser.LocalId = user.LocalId;
        GameManager.ActualUser.School = user.School;
        GameManager.ActualUser.Lessons = new bool[10];
        for (int i = 0; i < user.Lessons.Length; i++)
        {
            GameManager.ActualUser.Lessons[i] =
user.Lessons[i];
        }
    }
}
var les = GameManager.ActualUser.Lessons;

for (int i = 0; i < les.Length; i++)
{
    if (les[i] == true)
    {
        Lessons[i].SetActive(false);
    }
}

Panel.DOMoveX(540, 0.25f);
}

public void LogIn (string email, string password)
{
    string                userData                =
"{\"email\": \"\"+email+\"\", \"password\": \"\"+password+\"\", \"returnSecureToken\": true}";
    GameManager.Email = email;
    GameManager.Password = password;

    RestClient.Post<SignResponse>(LogInURL + AuthKey,
userData).Then(
    response =>
    {
        localId = response.localId;
        idToken = response.idToken;
        GameManager.ActualUser = GetUserName(localId);
        DOTween.Sequence()
            .Append(LogInMenu.transform.DOMoveX(2000,
0.25f))
            .Append(UserMenu.transform.DOMoveX(540,
0.25f));

    }).Catch(error =>
    {

```

```

        errorTextLogIn.gameObject.SetActive(true);
        errorTextLogIn.text = "Ошибка входа, проверьте данные";
    });

}

public User GetUserName(string localId)
{

    RestClient.Get(GetUsersURL).Then(
        response =>
        {
            fsData userData = fsJsonParser.Parse(response.Text);
            Debug.Log(response.Text);
            resp = response.Text;
            Dictionary<string, User> users = null;
            _serializer.TryDeserialize(userData, ref users);
            foreach (var user in users.Values)
            {
                Debug.Log(user.LocalId);
                if (user.LocalId.Equals(localId))
                {
                    flag.UserName = user.UserName;
                    flag.Class = user.Class;
                    flag.LocalId = user.LocalId;
                    flag.School = user.School;
                }
            }
        });

    return flag;
}
}

```

Листинг А.2 – Файл DownloadProvider.cs

```

using System.Collections;
using System.Collections.Generic;
using System.IO;
using DefaultNamespace.ScriptableObject;
using Firebase.Auth;
using Firebase.Storage;
using UnityEngine;
using System.Text.Json;
using System.Text.Json.Serialization;
using DG.Tweening;
using Newtonsoft.Json;
using Unity.VisualScripting;
using UnityEngine.Networking;
using Newtonsoft.Json.Converters;
using TMPPro;
using UnityEngine.SceneManagement;
using JsonConverter = Newtonsoft.Json.JsonConverter;
using JsonSerializer = Newtonsoft.Json.JsonSerializer;

```

```

namespace DefaultNamespace.Auth
{
    public class DownloadProvider : MonoBehaviour
    {
        static public Lesson CurrentLesson;
        [SerializeField] private TMP_Text Description;
        [SerializeField] private Transform MenuInfo;
        public string LessonName;
        public string Target;

        private void Start()
        {
            DontDestroyOnLoad(this);
        }

        public void SetLessonName(string name)
        {
            LessonName = name;
        }

        public void SetDescription(string description)
        {
            Description.text = description;
            MenuInfo.transform.DOMoveX(540, 0.25f);
        }

        public void SetTarget(string target)
        {
            Target = target;
        }

        public void DownloadLesson()
        {
            StartCoroutine(LoadLesson(LessonName));
        }

        IEnumerator LoadLesson(string lessonName)
        {
            var storage = FirebaseStorage.DefaultInstance;
            var someJson =

storage.GetReference($"{"/Lesson1/{lessonName}.json"});
            var downloadTask =
someJson.GetBytesAsync(long.MaxValue);
            yield return new WaitUntil(() =>
downloadTask.IsCompleted);
            Lesson some = null;
            var jsonString =
System.Text.Encoding.UTF8.GetString(downloadTask.Result);
            string str = jsonString.Substring(1, jsonString.Length-
1);

            some = JsonConvert.DeserializeObject<Lesson>(str);
            CurrentLesson = some;
            GetComponent<GameManager>().Lesson = some;
            SceneManager.LoadScene("new");
        }
    }
}

```

```

        }
        /*public void load()
        {
            FirebaseStorage storage =
            FirebaseStorage.DefaultInstance;
        }*/
    }
}

```

Листинг А.3 – Файл FirebaseInit.cs

```

using Firebase;
using TMPPro;
using UnityEngine;
using UnityEngine.Events;

namespace DefaultNamespace.Auth
{
    public class FirebaseInit : MonoBehaviour
    {
        public UnityEvent OnFirebaseInitialized = new UnityEvent();

        private async void Start()
        {
            var dependencyStatus = await
            FirebaseApp.CheckAndFixDependenciesAsync();
            if (dependencyStatus == DependencyStatus.Available)
            {
                Debug.Log("some");
            }
        }
    }
}

```

Листинг А.4 – Файл SignResponse.cs

```

using System;

[Serializable]
public class SignResponse
{
    public string localId;
    public string idToken;
}

```

Листинг А.5 – Файл User.cs

```

using System;

[Serializable]
public class User
{
    public string Class;
    public bool[] Lessons;
    public string LocalId;
    public string School;
    public string UserName;
}

```

```

        public User(string name, string localId, string school, string
Class)
    {
        Lessons = new bool[10];
        UserName = name;
        LocalId = localId;
        School = school;
        this.Class = Class;
        for (int i = 0; i < Lessons.Length; i++)
        {
            Lessons[i] = false;
        }
    }
    public User()
    {

    }
}

```

Листинг А.6 – Файл Colba.cs

```

using System.Collections;
using System.Collections.Generic;
using DefaultNamespace;
using DefaultNamespace.Auth;
using Proyecto26;
using UnityEngine;
using UnityEngine.SceneManagement;

namespace Display
{
    public class Colba : MonoBehaviour, IIntractable
    {
        public List<Items> reagents;
        public ParticleSystem ParticleSystem;

        public void AddReagent(Items reagent)
        {
            reagents.Add(reagent);
        }

        public bool CheckResult(List<string> itemsList)
        {
            int correct = 0;
            foreach (var item in itemsList)
            {
                if (reagents.Exists(x => x.Name == item))
                {
                    correct++;
                }
                else correct--;
            }

            if (correct == itemsList.Count)
            {
                return true;
            }
        }
    }
}

```

```

        return false;
    }

    public void Interact()
    {
        //Debug.Log(AllServices.Container.Single<UIService>().GetComponent<GameMa
nager>().Lesson.ActualItemsList);
        //List<string> some =
AllServices.Container.Single<UIService>().GetComponent<GameManager>().Les
son.ActualItemsList;
        List<string> some =
DownloadProvider.CurrentLesson.ActualItemsList;
        if (CheckResult(some))
        {
            User user =
AllServices.Container.Single<GameManager>().ActualUser;
            user.Lessons[0] = true;

            RestClient.Put(DataBaseInfo.databaseURL+"Users/"+user.LocalId+".json",
                user);

            AllServices.Container.Single<UIService>().EndLesssonMenu();
            StartCoroutine(EndLesson());
        }
    }

    public IEnumerator EndLesson()
    {
        ParticleSystem.Play();
        yield return new WaitForSeconds(10);
        SceneManager.LoadScene("StartScene");
    }
}

```

Листинг А.7 – Файл Game.cs

```

using DefaultNamespace;
using TMPPro;
using Unity.Mathematics;
using UnityEngine;
using UnityEngine.XR.ARFoundation;

namespace Display
{
    public class Game : MonoBehaviour
    {
        public User actualUser;
        private void Start()
        {
            AllServices.Container.RegisterSingle<IResourcesService>(new
ResourcesService());
        }
    }
}

```

```

AllServices.Container.Single<IResourcesService>().ARRaycastManager
GetComponent<ARRaycastManager>();

GetComponent<TargetFactory>().InstantiateObject(Vector3.zero);
    }
}

```

Листинг А.8 – Файл SelectItemService.cs

```

using System;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;

namespace Display
{
    public class SelectItemService : MonoBehaviour, IService
    {
        [SerializeField] private ARRaycastManager _arRaycastManager;

        private void Start()
        {
            AllServices.Container.RegisterSingle<SelectItemService>(this);
            this.enabled = false;
        }

        private void Update()
        {
            if (Input.touchCount > 0)
            {
                List<ARRaycastHit> hits = new List<ARRaycastHit>();
                Vector2 pos = Input.GetTouch(0).position;
                _arRaycastManager.Raycast(pos, hits,
TrackableType.Planes);
                RaycastHit hit;
                if (hits.Count == 1)
                {
                    if (Physics.SphereCast(hits[0].pose.position,
0.005f, hits[0].pose.forward, out hit))
                    {
                        GameObject chemicat =
hit.collider.gameObject;

                        AllServices.Container.Single<UIService>().InteractWithObjectMenu(chemicat
);

                        Gizmos.color = Color.green;
                        Gizmos.DrawSphere(hits[0].pose.position,
0.005f);

                        this.enabled = false;
                    }
                }
            }
        }
    }
}

```

```

    }
}

```

Листинг A.9 – Файл TargetMovement.cs

```

using System;
using System.Collections.Generic;
using TMPPro;
using Unity.VisualScripting;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;

namespace Display
{
    public class TargetMovement : MonoBehaviour, IService
    {
        private ARRaycastManager _arRaycastManager;

        private void Start()
        {
            _arRaycastManager =
AllServices.Container.Single<IResourcesService>().ARRaycastManager;

AllServices.Container.Single<IResourcesService>().Target = gameObject;
            gameObject.SetActive(false);
        }

        private void Update()
        {
            ShowMarker();
        }

        private void ShowMarker()
        {
            List<ARRaycastHit> hits = new List<ARRaycastHit>();
            _arRaycastManager.Raycast(new Vector2(Screen.width / 2,
Screen.height / 2), hits,
                TrackableType.Planes);
            if (hits.Count > 0)
            {
                transform.position = hits[0].pose.position;
            }
        }
    }
}

```

Листинг A.10 – Файл UIService.cs

```

using System;
using System.Collections.Generic;
using DefaultNamespace;
using DefaultNamespace.Auth;
using DefaultNamespace.ScriptableObjects;
using DG.Tweening;
using TMPPro;
using UnityEditor;
using UnityEngine;

```

```

namespace Display
{
    public class UIService : MonoBehaviour, IUIService
    {
        [SerializeField] private List<Items> firstLesson;
        [SerializeField] private List<Items> listOfObjects;
        [SerializeField] private TMP_Dropdown _dropdown;
        [SerializeField] private GameObject _optionMenu;
        [SerializeField] private GameObject _selectItemMenu;
        [SerializeField] private GameObject _spawnObjectMenu;
        [SerializeField] private GameObject _objectInfoMenu;
        [SerializeField] private GameObject _chemicatMenu;
        [SerializeField] private GameObject _mainMenu;
        [SerializeField] private TMP_Text _chemicatName;
        [SerializeField] private TMP_Text _chemicatInfo;
        [SerializeField] private TMP_Text _targetText;

        private Chemicat some;

        private GameObject colba;
        [SerializeField] private GameObject _interactableObject;

        [SerializeField] private GameObject _colbaMenu;

        public GameObject OptionMenu { get=>_optionMenu;
            private set { }
        }
        public GameObject SelectItemMenu { get=>_selectItemMenu;
            private set { }
        }
        public GameObject SpawnObjectMenu { get=>_spawnObjectMenu;
            private set { }
        }

        private void Start()
        {
            AllServices.Container.RegisterSingle<UIService>(this);
            foreach (var items in
DownloadProvider.CurrentLesson.AllItemsList)
            {
                _dropdown.options.Add(new TMP_Dropdown.OptionData()
{text = items});
            }

            _targetText.text =
AllServices.Container.Single<GameManager>().GetComponent<DownloadProvider
>().Target;

        }

        /// <summary>
        /// Buttons
        /// </summary>
        public void SelectItemON()
        {

```

```

AllServices.Container.Single<SelectItemService>().enabled = true;
    }

    public void TargetOn()
    {

AllServices.Container.Single<IResourcesService>().Target.SetActive(true);
    }

    public void TargetOff()
    {

AllServices.Container.Single<IResourcesService>().Target.SetActive(false)
;
    }
    public void DeleteInteractableObject()
    {
        Destroy(_interactableObject);
    }

    public void AddChemicatToColba()
    {
        Items reog = new Items();
        reog.Name =
        _interactableObject.GetComponent<ChemicatAdapter>().Name;
        colba.GetComponent<Colba>().AddReogent(reog);
        DOTween.Sequence()
            .Append(_interactableObject.transform.DOMove(new
Vector3(_interactableObject.transform.position.x,
        _interactableObject.transform.position.y
+
transform.localScale.y / 3,
        _interactableObject.transform.position.z),
0.5f))
            .Append(_interactableObject.transform.DOMove(new
Vector3(
        colba.transform.position.x
+
_interactableObject.transform.localScale.x / 5,
        _interactableObject.transform.position.y
+
transform.localScale.y / 3,
        colba.transform.position.z), 0.5f))
            .Append(_interactableObject.transform.DORotate(new
Vector3(0, 0, 110), 1))
            .Append(_interactableObject.transform.DOScale(0,
1)).OnComplete(() =>
        {
            DeleteInteractableObject();
        });

        //DeleteInteractableObject();
    }
    public void Interact()
    {

_interactableObject.GetComponent<IIntractable>().Interact();
        if (_interactableObject.GetComponent<Chemicat>())
        {
            _mainMenu.SetActive(true);

```

```

    }

    }
    public void Spawn(TMP_Dropdown dropdown)
    {
        if (dropdown.value == 0)
        {
            colba =
AllServices.Container.Single<ColbaFactory>().InstantiateObject(AllService
s.Container.Single<IResourcesService>().Target.transform.position);

        }
        else
        {
            GameObject pref = firstLesson.Find(x => x.Name ==
_dropdown.options[_dropdown.value].text).Prefab;
            GameObject reg
=AllServices.Container.Single<ChemicatFactory>().InstantiateObject(AllSer
vices.Container.Single<IResourcesService>().Target.transform.position,
pref);

            reg.GetComponent<Chemicat>().Name =
            firstLesson.Find(x => x.Name ==
_dropdown.options[_dropdown.value].text).Name;

            _dropdown.options.Remove(_dropdown.options[_dropdown.value]);
        }
        TargetOff();

    }
    /// <summary>
    /// Actions with interactable object
    /// </summary>
    /// <param name="interactableObject"></param>
    public void InteractWithObjectMenu(GameObject
interactableObject)
    {
        //_objectInfoMenu.SetActive(true);
        _interactableObject = interactableObject;
        OpenInteractableMenu(_interactableObject);

    }

    private bool OpenInteractableMenu(GameObject
interactableObject)
    {
        if (interactableObject.GetComponent<Chemicat>())
        {
            _chemicatMenu.SetActive(true);
            _chemicatName.text =
interactableObject.GetComponent<ChemicatAdapter>().Name;
            //_chemicatInfo.text =
interactableObject.GetComponent<ChemicatAdapter>().Info;

            return true;
        }
        else if (interactableObject.GetComponent<Colba>())
        {
            _colbaMenu.SetActive(true);

```

```

        return true;
    }
    return false;
}

public void EndLesssonMenu()
{
    _mainMenu.SetActive(false);
    _targetText.text = "Поздравляем, вы прошли урок!";
}

}
};

```

Листинг А.11 – Файл AllServices.cs

```

using UnityEngine;

namespace Display
{
    public class AllServices : MonoBehaviour
    {
        private static AllServices _instance;
        public static AllServices Container =>
            _instance ?? (_instance = new AllServices());

        public void RegisterSingle<TService>(TService
implementation) where TService : IService =>
            Implementation<TService>.ServiceInstance =
implementation;

        public TService Single<TService>() where TService:IService
=>
            Implementation<TService>.ServiceInstance;

        private static class Implementation<TService> where TService
: IService
        {
            public static TService ServiceInstance;
        }
    }
}

```

Листинг А.12 – Файл ChemicatFactory.cs

```

using System;
using UnityEngine;

namespace Display
{
    public class ChemicatFactory : InstantiateFactory
    {
        private void Start()
        {
            AllServices.Container.RegisterSingle<ChemicatFactory>(this);
        }
    }
}

```

Листинг A.13 – Файл ColbaFactory.cs

```
using UnityEngine;

namespace Display
{
    public class ColbaFactory : InstantiateFactory
    {
        private void Start()
        {
            AllServices.Container.RegisterSingle<ColbaFactory>(this);
        }
    }
}
```

Листинг A.14 – Файл InitializeGameData.cs

```
using System.Collections;
using System.Collections.Generic;
using Display;
using UnityEngine;

public class InitializeGameData : MonoBehaviour
{
    void Start()
    {
        AllServices.Container.RegisterSingle<IResourcesService>(new
ResourcesService());

        GetComponent<TargetFactory>().InstantiateObject(Vector3.zero);
    }

    void Update()
    {
    }
}
```

Листинг A.15 – Файл InstantiateFactory.cs

```
using System;
using DefaultNamespace;
using Unity.Mathematics;
using UnityEngine;
using UnityEngine.UIElements;

namespace Display
{
    public class InstantiateFactory : MonoBehaviour, IService
    {
        [SerializeField] protected GameObject _prefab;

        public GameObject InstantiateObject(Vector3 position)
        {
            return Instantiate(_prefab, position,
                quaternion.identity);
        }
    }
}
```

```

        public GameObject InstantiateObject(Vector3 position,
        GameObject pref)
        {
            return Instantiate(pref, position,
                               quaternion.identity);
        }
    }
};

```

Листинг A.16 – Файл Items.cs

```

using UnityEngine;

namespace Display
{
    [CreateAssetMenu(menuName = "Create Items", fileName = "Items",
order = 0)]
    public class Items : ScriptableObject
    {
        public string Name;
        public GameObject Prefab;
        public bool isChemicat;
    }
}

```

Листинг A.17 – Файл Lesson.cs

```

using System;
using System.Collections.Generic;
using System.Text.Json.Serialization;
using Display;
using UnityEngine;
using UnityEngine.Serialization;

namespace DefaultNamespace.ScriptableObjects
{
    [Serializable]
    public class Lesson
    {
        [FormerlySerializedAs("List")] [JsonPropertyName("List")]
        public List<string> ActualItemsList;
        [JsonPropertyName("AllItemList")]
        public List<string> AllItemsList;

        [JsonConstructor]
        public Lesson(List<string> actualItemsList, List<string>
allItemsList)
        {
            ActualItemsList = actualItemsList;
            AllItemsList = allItemsList;
        }
    }
}

```

Листинг A.18 – Файл Chemicat.cs

```

using UnityEngine;

```

```

namespace DefaultNamespace
{
    public class Chemicat : MonoBehaviour
    {
        private string _name;
        private string _info;
        public string Name
        {
            get => _name;
            set { _name = value; }
        }
        public string Info
        {
            get => _info;
            set { _info = value; }
        }
    }
}

```

Листинг A.19 – Файл ChemicatAdapter.cs

```

using Display;
using UnityEngine;

namespace DefaultNamespace
{
    public class ChemicatAdapter : Chemicat, IIntractable
    {
        public void Interact()
        {
            AllServices.Container.Single<UIService>().AddChemicatToColba();
        }
    }
}

```

Листинг A.20 – Файл GameManager.cs

```

using System;
using System.Collections;
using System.Collections.Generic;
using DefaultNamespace.ScriptableObjects;
using Display;
using UnityEngine;

public class GameManager : MonoBehaviour, IService
{
    public User ActualUser;
    public string Email;
    public string Password;
    public Lesson Lesson;

    public void Start()
    {
        if (AllServices.Container.Single<GameManager>() == null)
        {
            AllServices.Container.RegisterSingle<GameManager>(this);
        }
    }
}

```

```

    }
}

```

Листинг A.21 – Файл IIntractable.cs

```

namespace DefaultNamespace
{
    public interface IIntractable
    {
        public void Interact();
    }
}

```

Листинг A.22 – Файл IResourcesService.cs

```

using UnityEngine;
using UnityEngine.XR.ARFoundation;

namespace Display
{
    public interface IResourcesService : IService
    {
        public ARRaycastManager ARRaycastManager { get; set; }
        public GameObject Target { get; set; }
    }
}

```

Листинг A.23 – Файл Authentication.cs

```

using UnityEngine;

namespace Display
{
    public interface IUIService : IService
    {
        public void InteractWithObjectMenu(GameObject some);
    }
}

```

Листинг A.24 – Файл Main.cs

```

using System;
using System.Collections.Generic;
using DefaultNamespace;
using DefaultNamespace.Auth;
using DefaultNamespace.ScriptableObjects;
using TMPPro;
using UnityEditor;
using UnityEngine;
namespace Display
{
    public class UIService : MonoBehaviour, IUIService
    {
        [SerializeField] private List<Items> firstLesson;
        [SerializeField] private List<Items> listOfObjects;
        [SerializeField] private TMP_Dropdown _dropdown;
        [SerializeField] private GameObject _optionMenu;
        [SerializeField] private GameObject _selectItemMenu;
        [SerializeField] private GameObject _spawnObjectMenu;
    }
}

```

```

[SerializeField] private GameObject _objectInfoMenu;
[SerializeField] private GameObject _chemicatMenu;
[SerializeField] private TMP_Text _chemicatName;
[SerializeField] private TMP_Text _chemicatInfo;
private GameObject _interactableObject;
[SerializeField] private GameObject _colbaMenu;

public GameObject OptionMenu { get=>_optionMenu;
    private set { }
}
public GameObject SelectItemMenu { get=>_selectItemMenu;
    private set { }
}
public GameObject SpawnObjectMenu { get=>_spawnObjectMenu;
    private set { }
}
private void Start()
{
    AllServices.Container.RegisterSingle<IUIService>(this);
    foreach (var items in
DownloadProvider.CurrentLesson.List)
    {
        if (firstLesson.Exists(x => x.Name.Equals(items)))
        {
            _dropdown.options.Add(new
TMP_Dropdown.OptionData() {text = items});
        }
    }
    /// <summary>
    /// Buttons
    /// </summary>
    public void SelectItemON()
    {

AllServices.Container.Single<SelectItemService>().enabled = true;
    }
    public void TargetOn()
    {

AllServices.Container.Single<IResourcesService>().Target.SetActive(true);
    }
    public void TargetOff()
    {

AllServices.Container.Single<IResourcesService>().Target.SetActive(false)
;
    }
    public void DeleteInteractableObject()
    {
        Destroy(_interactableObject);
    }
    public void Interact()
    {

_interactableObject.GetComponent<IIntractable>().Interact();
    }
    public void Spawn(TMP_Dropdown dropdown)

```

```

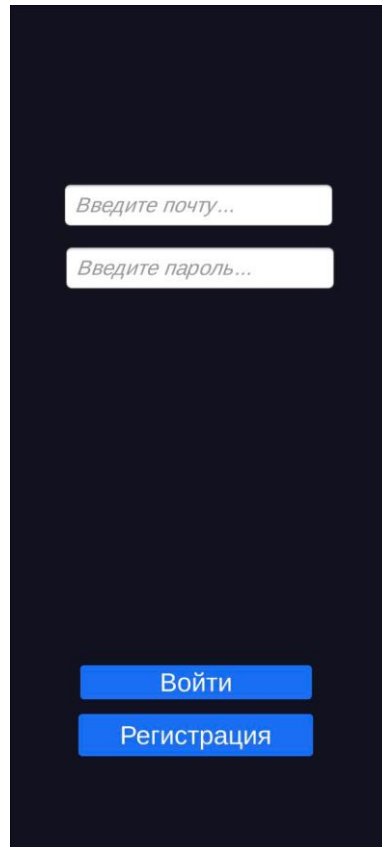
        {
            if (dropdown.value == 0)
            {
                AllServices.Container.Single<ColbaFactory>().InstantiateObject(AllServices.Container.Single<IResourcesService>().Target.transform.position);
            }
            else if (dropdown.value == 1)
            {
                AllServices.Container.Single<ChemicatFactory>().InstantiateObject(AllServices.Container.Single<IResourcesService>().Target.transform.position);
            }
            AllServices.Container.Single<UIService>().TargetOff();
        }
        /// <summary>
        /// Actions with interactable object
        /// </summary>
        /// <param name="interactableObject"></param>
        public void InteractWithObjectMenu(GameObject interactableObject)
        {
            //_objectInfoMenu.SetActive(true);
            _interactableObject = interactableObject;
            OpenInteractableMenu(_interactableObject);
        }
        private bool OpenInteractableMenu(GameObject interactableObject)
        {
            if (interactableObject.GetComponent<Chemicat>())
            {
                _chemicatMenu.SetActive(true);
                _chemicatName.text = interactableObject.GetComponent<ChemicatAdapter>().Name;
                //_chemicatInfo.text = interactableObject.GetComponent<ChemicatAdapter>().Info;
                return true;
            }
            else if (interactableObject.GetComponent<Colba>())
            {
                _colbaMenu.SetActive(true);
                return true;
            }
            return false;
        }
    }
}

```

Приложение А

(рекомендуемое)

Результаты реализации интерфейса



Введите почту...

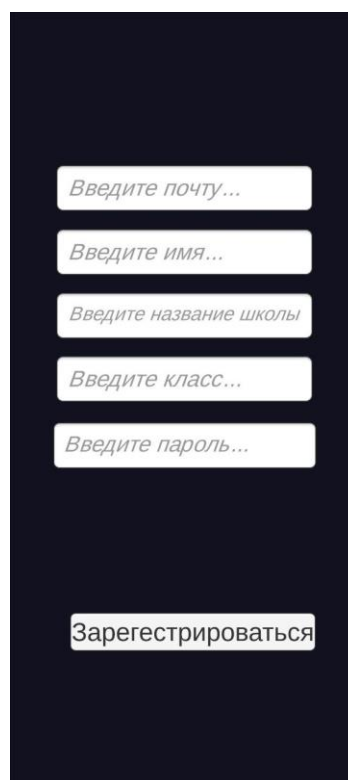
Введите пароль...

Войти

Регистрация

This is a login interface mockup on a dark blue background. It features two white input fields at the top, one for email (labeled 'Введите почту...') and one for password (labeled 'Введите пароль...'). Below these fields are two blue buttons: 'Войти' (Login) and 'Регистрация' (Registration).

Рисунок Б.1 – Интерфейс страницы входа



Введите почту...

Введите имя...

Введите название школы

Введите класс...

Введите пароль...

Зарегистрироваться

This is a registration interface mockup on a dark blue background. It features five white input fields stacked vertically, labeled 'Введите почту...', 'Введите имя...', 'Введите название школы', 'Введите класс...', and 'Введите пароль...'. At the bottom is a white button labeled 'Зарегистрироваться' (Register).

Рисунок Б.2 – Интерфейс страницы регистрации

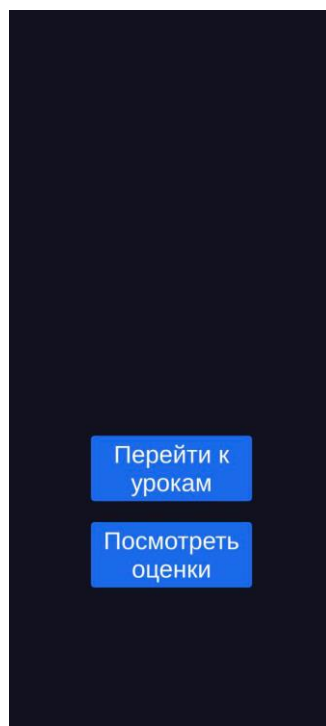


Рисунок Б.3 – Интерфейс страницы переходов

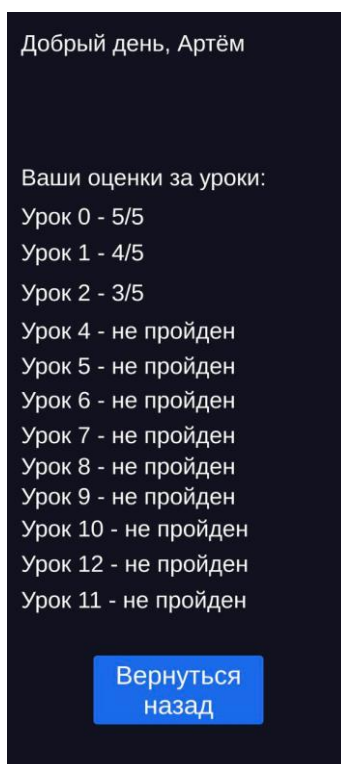


Рисунок Б.4 – Интерфейс страницы информации ученика

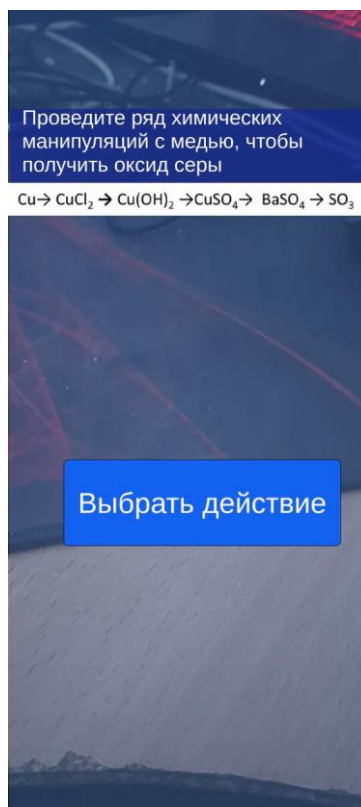


Рисунок Б.5 – Интерфейс выбора действий

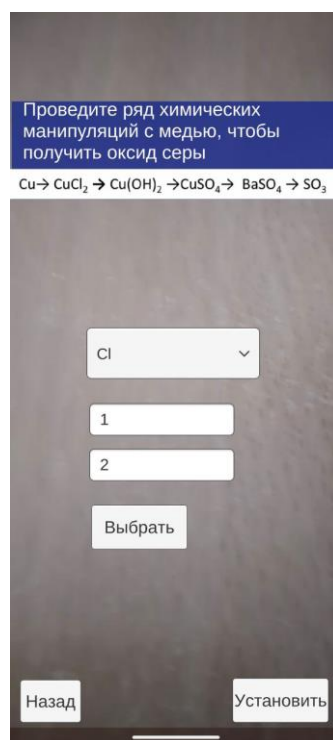


Рисунок Б.6 – Интерфейс окна с установкой объектов



Рисунок Б.7 – Интерфейс окна с выбором установленных объектов

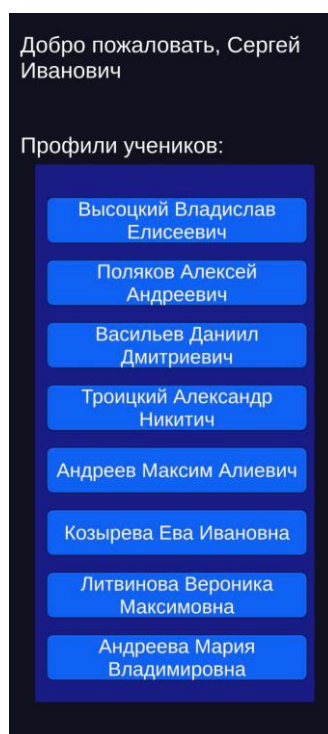


Рисунок Б.8 – Интерфейс страницы со списком учеников



Рисунок Б.9 – Интерфейс страницы выбора урока



Рисунок Б.10 – Интерфейс страницы информации об уроке

Приложение В
(рекомендуемое)
Таблица диаграммы класса

Таблица 14 – Спецификация классов

Класс	Описание класса	Поля	Описание полей	Методы	Описание методов
Authentication	Класс аутентификации и регистрации и пользователя	User	Вошедший пользователь	Start()	Начальный метод для инициализации программ
		AuthKey	Ключ для работы с Firebase Authentication	PostToDatabase(string localid, string name, string school, string class)	Метод для записи в базу данных
		DatabaseURL	Ссылка для Rest запросов к Firebase Authentication	RegisterIn(string localid, string name, string school, string class)	Метод для отправки REST запросов регистрации к Firebase Authentication
		RegisterURL	Ссылка для Rest запросов регистрации к Firebase Authentication	Login()	Метод для отправки REST запросов авторизации пользователя к Firebase Authentication
		LoginURL	Ссылка для Rest запросов авторизации к Firebase Authentication		
		GetUserURL	Ссылка для Rest запросов получения пользователя по localId к Firebase Authentication		
MonoBehavior	Класс Unity для работы платформы	SceneManager	Управление сценами в Unity	Update()	Метод для отображения изменений
				Start()	Начальный метод для инициализации программ
DownloadProvider	Класс для скачивания текущего урока	Lesson	Урок, скачанный с облачного хранилища	Start()	Начальный метод для инициализации программы
				LoadLesson(string lessonName)	Метод для скачивания

Продолжение таблицы 14

Класс	Описание класса	Поля	Описание полей	Методы	Описание методов
Lesson	Класс заготовки урока	ActualItemsList[]	Список компоненто в необходимых для успешного выполнения урока		
		AllItemsList[]	Список всех компоненто в на урок		
Items	Класс компонент для взаимодействия	Name	Имя компонента		
		Prefab	Готовая модель с прикрепленными компонентами		
		isChemicat	Является ли объект реагентом		
GameObject	Класс Unity, с которым можно взаимодействовать на сцене			GetComponent()	Метод для получения компонента объекта
				SetActive()	Метод отключения и включения объекта на сцене
Chemicat	Name	Имя реагента			
	Info	Информация, которую необходимо отобразить на экране			
UIService	Класс, для взаимодействия с пользователем	InteractableObject	Объект, с которым взаимодействует ученик	Start()	Начальный метод для инициализации и программы
				TargetOn()	Метод для включения маркера с местом установки объекта
				TargetOff()	Метод для отключения маркера

Окончание таблицы 14

Класс	Описание класса	Поля	Описание полей	Методы	Описание методов
				AddChemicalToColba()	Добавление реагента или элемента, выбранного пользователем в колбу
				Interact()	Взаимодействие с интерактивным объектом
				Spawn()	Установка объекта
				EndLesson()	Завершение урока и отправка результатов в БД